Istituto di Istruzione Secondaria Superiore Galileo Galilei

Istituto tecnico tecnologico

Indirizzo di Informatica e Telecomunicazioni

PROGETTO NOLEGGIO BICI

"BIKE SHARING"

Creazione di un servizio di noleggio bici secondo le richieste della prima simulazione della seconda prova per indirizzo informatica e telecomunicazioni.

Presentata da:
STEFANO COSTA
HANNES MAIR

Anno accademico 2018-2019

Indice

DESCRIZIONE DEL PROGETTO 6
LISTA SERVIZI RICHIESTI 6
SERVIZI PER GLI UTENTI 6
- Registrazione e autenticazione al sito web 6
- Controllo e gestione dei propri noleggi 6
- Controllo delle stazioni di noleggio 6
- Noleggio delle bici 6
- Ricarica del saldo 6

SERVIZI PER IL GESTORE 7
- Visualizzazione delle statistiche 7
- Visualizzazione dello stato delle bici e delle stazioni 7
- Modifica immediata ai servizi 7

PROGETTAZIONE 8
SERVER WEB E DATABASE 8
PROGETTAZIONE DEL SITO 8
PROGETTAZIONE SIMULATORE DI BICI E STAZIONI 9
PROGETTAZIONE DATABASE 10
CODICE SQL 11
Tabella “BICI” 11
Tabella “NOLEGGIA” 11
Tabella “STAZIONI” 12
Tabella “UTENTI” 12
STRUTTURA DEL SITO 13
Introduzione 13
File Ausiliari 13
- File di index 13
- File di configurazione 16
- File di trilinguismo 16
- File di header 16
- File di policy 18
- File per le timbrature 18

Pagine Web 19
- Home 19
- Sezione “come funziona” 19
- Sezione “stazioni” 20
- Sezione “noleggi” 20
- Register 21
- Login 22
- Sezione “user” 23
- Logout 24
INSTALLAZIONE DEL SERVER WEB E DEL DATABASE 25
Apache 25
PHP7.0 25
MySQL 25
PhpMyAdmin 25
REALIZZAZIONE SIMULATORE DI STAZIONE 26
INTRODUZIONE 26
ACCENNI TEORICI SUL RFID 26
ASSEMBLAGGIO DEL SIMULATORE 27
- LCD 28
- LED 28
- Pulsanti 28
- Lettore 28
- Involucro di plastica 28
FASI DI FUNZIONAMENTO 30
- Lettura del badge 30
- Proposta delle bici 30
- Inizio noleggio 30
- Termine noleggio 30
ACCENNI ALLA PROGRAMMAZIONE PYTHON 31
ANALISI DEL CODICE NELLE FASI DI FUNZIONAMENTO 32
Lettura del badge e controllo nel database 32
Controllo delle bici nel database 32
Inizio noleggio e uso della bici 33
Termine del noleggio 33
RIFERIMENTI E FONTI 34

DESCRIZIONE DEL PROGETTO

Il Comune di una città europea di medie dimensioni vuole implementare un servizio di noleggio di biciclette attraverso stazioni di "noleggio e consegna" dislocate in diversi punti della città. Al fine di addebitare il costo del servizio, si vuole conoscere in ogni momento chi ha preso in uso una determinata bici. Il servizio è fruibile previa registrazione online dei dati dell'utente, a seguito della quale il Comune provvederà alla consegna di una tessera elettronica che conterrà l'identificativo dell'utente leggibile in modalità *contactless*.

Per noleggiare una bicicletta l'utente dovrà avvicinare la propria tessera ad un apposito lettore, unico per stazione, di conseguenza verrà sbloccata una delle biciclette (scelta dall'utente). L'utente potrà successivamente riconsegnare la bicicletta in qualsiasi stazione abbia uno slot libero.

LISTA SERVIZI RICHIESTI

Dalle richieste emergono i servizi che la soluzione deve essere in grado di offrire:

SERVIZI PER GLI UTENTI

Ognuno dei servizi elencati sarà fruibile in 3 lingue (supponendo di essere in Trentino Alto Adige): Italiano, tedesco e inglese.

- Registrazione e autenticazione al sito web

  La registrazione al sito è obbligatoria per usufruire del servizio di noleggio. Esso serve agli utenti registrati per accedere ad azioni avanzate nella gestione del proprio account e dei propri noleggi.

- Controllo e gestione dei propri noleggi

  Mantenere sotto controllo tutti i noleggi con relativi dati tra cui il costo e la durata. Possibilità di gestire il pagamento dei noleggi.

- Controllo delle stazioni di noleggio

  Utilizzare un'applicazione che mostri su una mappa le varie stazioni di noleggio con i relativi slot (liberi e occupati) con relative features quali indicare la strada alla stazione più vicina con almeno una bici o prenotare una bici ad una stazione.

- Noleggio delle bici

  Usando il proprio badge personale, gli utenti potranno prenotare una bici con la possibilità di sceglierla, di modo che si possano noleggiare bici da bambino e da adulto.

- Ricarica del saldo

  Utilizzare il sito per ricaricare il proprio saldo con una carta di credito, dal conto corrente o affini.

SERVIZI PER IL GESTORE

- Visualizzazione delle statistiche

  Un sito web o un'applicazione dedicata avrà il compito di mostrare le varie statistiche nei noleggi, nei guadagni e nell'utenza per poter agire rapidamente in base all'andamento dei servizi.

- Visualizzazione dello stato delle bici e delle stazioni

  Non essendo dati ricavabili automaticamente, lo stato delle bici e delle stazioni deve essere mantenuto sotto controllo attraverso delle telecamere installate in ogni stazioni e attraverso un servizio di rating del noleggio, hostato sui lettori nelle stazioni: questo consentirà agli utenti di segnalare un malfunzionamento nelle bici o nei lucchetti delle stazioni offrendo loro, per invogliarli a partecipare a questo servizio, un compenso nel personale saldo.

- Modifica immediata ai servizi

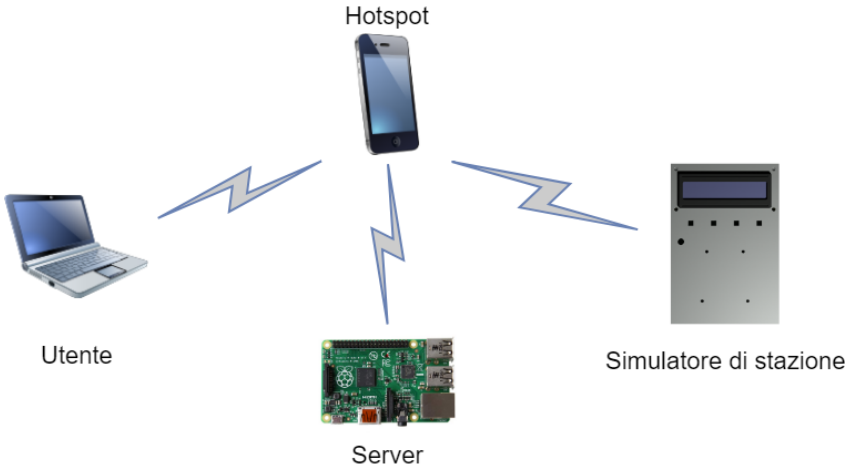
  Attraverso un software o un sito, la modifica di dati sul database e non sarà immediata e non necessiterà una profonda conoscenza dell'informatica. Sarà quindi possibile modificare i costi, bloccare alcuni lucchetti, sospendere account dell'utenza...

PROGETTAZIONE

Dalla lista dei servizi richiesti è chiaro che ideare e a imbastire una soluzione completa sia oltre le nostre possibilità di tempo e di competenza. Ci siamo quindi limitati a creare il sito web per gli utenti e a realizzare un lettore RFID che simulasse una stazione con 4 bici.

Dai requisiti il progetto necessita di: server web, database, sito web per l'utenza e simulatore di bici e di stazioni.

Non potendo adempiere a tutti i servizi, ci siamo limitati a hostare il server web e il database su un raspberry, simulare le bici e le stazioni su un altro raspberry (con adeguato hardware tra cui il lettore RFID, LED e LCD) e creare un sito web in HTML, PHP e Javascript.



SERVER WEB E DATABASE

Il server web e il database sono hostati su uno dei 2 raspberry.

Per face ciò è stato necessario installare Apache2, php7.0, mysql e phpmyadmin.

PROGETTAZIONE DEL SITO

Dalle richieste del progetto emergono i requisiti del sito visitato dagli utenti:

- Registrazione degli utenti
- Login degli utenti
- Gestione dei cookie per mantenere l'accesso
- Trilinguismo (supponendo di essere in Trentino-Alto Adige: Inglese, Tedesco, Italiano)
- Mostrare un listato di stazioni con relativi slot occupati, liberi, rotti (non viene implementata una mappa con cui interagire)
- Mostrare il listato di noleggi con tutti i dati necessari
- Mostrare il proprio account con il relativo saldo

Non potendo provvedere ad un servizio per la ricarica del saldo nell'account (considerati i problemi di sicurezza, la verifica sulle carte di pagamento e lo scopo didattico del progetto), il sito si limita a mostrare il saldo dell'utente.

PROGETTAZIONE SIMULATORE DI BICI E STAZIONI

Per simulare il progetto con le stazioni e il server centrale, sono stati utilizzati 2 raspberry: 1 con funzione di web server e di database e 1 per simulare il lettore badge apposto ad ogni stazione. Il lettore è stato realizzato con la tecnologia RFID (con appositi badge) e con un lettore LCD. Il tutto è stato inserito all'interno di una scatola prodotta con una stampante 3D per rendere realistica l'interazione di un utente. Al posto delle bici sono stati collegati dei pulsanti al raspberry, in modo che l'utente potesse simulare la scelta della bici premendo uno dei 4 pulsanti che simboleggia 4 diverse bici.

Il lettore RFID provvederà a:

- Leggere il badge e autenticare l'utente
- Iniziare un noleggio con una bici scelta dall'utente
- Terminare un noleggio calcolandone il costo
- Detrarre il costo del noleggio dal saldo dell'utente (se è sufficiente)

PROGETTAZIONE DATABASE

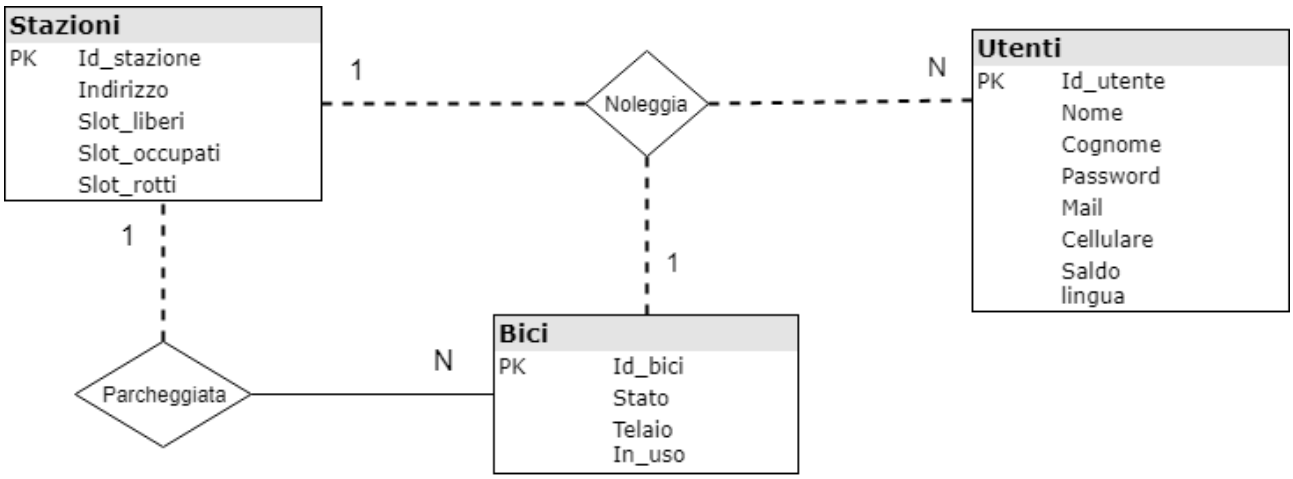


Tabelle derivate

Stazioni (Id_stazione, Indirizzo, Slot_liberi, Slot_occupati, Slot_rotti)

Bici(Id_bici, Id_stazione, Stato, Telaio, In_uso), Id_stazione FK

Utenti(Id_utente, Nome, Cognome, Password, Mail, Cellulare, Saldo, lingua)

Noleggia(Id_utente, Id_bici, Data_prestito, Data_consegna, Stazione_partenza, Stazione_arrivo, Costo, Pagato),
Id_utente, Id_bici, Stazione_partenza, Stazione_arrivo FK

Dai requisiti esposti risulta che il sistema debba gestire degli utenti, delle bici, delle stazioni e debba gestire il calcolo dei costi. È quindi chiaro che si necessiti di una tabella "Utenti", per tenere conto degli usufruttuari del sistema di noleggio con i loro dati personali tra cui "lingua", cioè la lingua che l'utente ha impostato sul sito all'ultimo accesso (ogni volta che un utente cambia lingua, viene modificato il campo "lingua", per, al successivo login, ripresentare il sito nella lingua scelta); una tabella "Stazioni", per segnalare le stazioni con i relativi slot liberi, occupati o rotti; una tabella "Bici", per identificare ogni bici nei relativi noleggi (così da tenere conto di quale utente usa quale bici), tenere conto dello stato della bici per far intervenire istantaneamente un servizio di manutenzione, memorizzare il telaio della bici in modo da poter, nel caso di furto, sporgere denuncia.

Dall'associazione di "Bici", "Stazioni" e "Utenti" risulta la tabella "Noleggia", che terrà memorizzati tutti i noleggi tenendo conto dell'utente, della bici, della stazione di provenienza e di arrivo e della durata del noleggio.

CODICE SQL

Tabella “BICI”

```
CREATE TABLE `Bici` (
  `Id_bici` int(3) NOT NULL PRIMARY KEY AUTO_INCREMENT,
  `Id_stazione` int(10) NOT NULL,
   FOREIGN KEY (`Id_stazione`) REFERENCES `Stazioni` (`Id_stazione`)
   ON DELETE CASCADE
   ON UPDATE CASCADE,
  `Stato` set('in servizio','in riparazione') NOT NULL,
  `Telaio` varchar(5) NOT NULL,
  `In_uso` tinyint(1) NOT NULL DEFAULT '0' COMMENT '0-> disponibile, 1-> in uso'
);
```

```
INSERT INTO `Bici` (`Id_bici`, `Id_stazione`, `Stato`, `Telaio`)
VALUES(1, 1, 'in servizio', 'js92k',0),
(2, 1, 'in servizio', 'k9dke',0),
(3,1, ’in servizio’, ‘ki222’, 0),
(4,1, ’in servizio’, ‘iopa2’,0);
```

Tabella “NOLEGGIA”

```
CREATE TABLE `Noleggia` (
  `Id_utente` bigint(10) UNSIGNED NOT NULL,
  `Id_bici` int(3) NOT NULL,
  `Data_prestito` datetime NOT NULL,
  `Data_consegna` datetime DEFAULT NULL,
  `Stazione_partenza` int(10) NOT NULL,
  `Stazione_arrivo` int(10) DEFAULT NULL,
  `Costo` float DEFAULT NULL,
  `Pagato` tinyint(1) DEFAULT '0',
   FOREIGN KEY (`Id_bici`) REFERENCES `Bici` (`Id_bici`)
   ON DELETE CASCADE
   ON UPDATE CASCADE,
   FOREIGN KEY (`Id_utente`) REFERENCES `Utenti` (`Id_utente`)
   ON DELETE CASCADE
   ON UPDATE CASCADE,
   FOREIGN KEY (`Stazione_arrivo`) REFERENCES `Stazioni` (`Id_stazione`)
   ON DELETE CASCADE
   ON UPDATE CASCADE,
   FOREIGN KEY (`Stazione_partenza`) REFERENCES `Stazioni` (`Id_stazione`)
   ON DELETE CASCADE
   ON UPDATE CASCADE,
   PRIMARY KEY (`Id_utente`,`Id_bici`,`Data_prestito`)
);
```

```
INSERT INTO `Noleggia` (`Id_utente`, `Id_bici`, `Data_prestito`, `Data_consegna`,
`Stazione_partenza`, `Stazione_arrivo`, `Costo`, `Pagato`) VALUES
(920234434229, 1, '2019-05-10 13:20:05', '2019-05-10 13:23:06', 1, 1, 0.20, 0),
(920234434229, 4, '2019-05-15 15:20:10', '2019-05-15 15:39:20', 1, 1, 0.80, 0),
(920234434229, 2, '2019-05-10 17:00:34', '2019-05-10 17:04:12', 1, 1, 0.10, 0);
```

Tabella “STAZIONI”

```
CREATE TABLE `Stazioni` (
  `Id_stazione` int(10) NOT NULL PRIMARY KEY AUTO_INCREMENT,
  `Indirizzo` varchar(255) NOT NULL,
  `Slot_liberi` int(10) NOT NULL,
  `Slot_occupati` int(10) NOT NULL,
  `Slot_rotti` int(10) NOT NULL
);
```

```
INSERT INTO `Stazioni` (`Id_stazione`, `Indirizzo`, `Slot_liberi`,
`Slot_occupati`, `Slot_rotti`)
VALUES(1, 'Piazza Vittoria', 10, 0, 0),
(2, 'Piazza Verdi', 10, 0, 0),
(3, 'Piazza Marconi', 10, 0, 0),
(4, 'Piazza Municipio', 10, 0, 0),
(5, 'Piazza Bruni', 9, 1, 0);
```

Tabella “UTENTI”

```
CREATE TABLE `Utenti` (
  `Id_utente` bigint(12) UNSIGNED NOT NULL PRIMARY KEY AUTO_INCREMENT,
  `Nome` varchar(255) NOT NULL,
  `Cognome` varchar(255) NOT NULL,
  `Password` varchar(255) NOT NULL,
  `Mail` varchar(255) NOT NULL UNIQUE,
  `Cellulare` varchar(10) NOT NULL,
  `Saldo` float NOT NULL,
  `lingua` varchar(3) NOT NULL DEFAULT 'ita'
);
```

```
INSERT INTO `Utenti` (`Id_utente`, `Nome`, `Cognome`, `Password`, `Mail`,
`Cellulare`, `Saldo`, `lingua`)
VALUES(920234434229, 'Hannes', 'Mair',
'b109f3bbbc244eb82441917ed06d618b9008dd09b3befd1b5e07394c706a8bb980b1d7785e5976ec
049b46df5f1326af5a2ea6d103fd07c95385ffab0cacbc86', 'hannes.mair2000@gmail.it',
'3332834020', 0, 'ita'),
(920234434230, 'Steve', 'Costa',
'fd522aae0ab5fe49b17ecdd09e90fe18c5ed6fb443f0c59283c4a06211b8837660e54d3bbc38c4a6
767b51432b634cc125b5482842c5dc1f4851c7c297c4c225', 'steve@gmail.com',
'3334468329', 0, 'ita'),
```

STRUTTURA DEL SITO

Introduzione

Per la creazione della parte statica del sito è stato utilizzato il software "Bootstrap Studio", consentendoci di creare una GUI (Graphical User Interface) in modo semplice ed efficace. Si sono in questo modo creati velocemente i form di login e registrazione, l'impostazione delle pagine (l'incolonnamento dei paragrafi, le impostazioni delle immagini...) e l'header.

Per quanto riguarda la grafica, quindi le immagini, gli sfondi e i vari stili, non si è dedicato molto tempo, in quanto la maggior parte è stato usato per la programmazione e la progettazione del Back End. È stata quindi adottata una grafica semplice, in stile "cartoon", con colori sgargianti e immagini semplici, il tutto a tema "stradale", in modo da rendere semplice e fruibile la navigazione da parte degli utenti.

Un fattore molto importante sono le immagini che, non avendo un budget da investire, sono tutte senza copyright, quindi utilizzabili gratuitamente, oppure create in autonomia.

Per quanto riguarda l'ambito database, è stato utilizzato "PhpMyAdmin" per la creazione del database in quanto semplice e veloce, e il PDO per interfacciarsi in modo sicuro e veloce al DBMS.

Infine, il sito ha una struttura modulare, cioè l'utente non accede direttamente alle pagine che visita, ma accede sempre alla pagina "index.php", che provvede a costruire ciò che l'utente vuole visualizzare includendo le altre pagine, che risultano come tasselli. Perciò l'utente passerà sempre dalla pagina di index, tenendo sotto controllo ogni azione dei visitatori del sito.

File Ausiliari

- File di index

Il sito è costruito con un approccio modulare: il file "index.php" gestisce l'inclusione di altri file .php per mostrare le varie pagine. Questo fa in modo che ogni attività sul sito passi attraverso l'index, obbligandoci ad inserire una parte di codice per la verifica dell'uso corretto del sito: gli utenti possono solo visualizzare il file "index.php", non possono accedere direttamente agli altri file. Per fare ciò, all'inizio di ogni file .php viene inserito il segmento di codice:

```php
$PHP_SELF = $_SERVER['PHP_SELF'];
//controlla che il path del file non sia quello richiamato nell'URL
if("/Bike_sharing/noleggi.php" == $PHP_SELF){
      header("Location: index.php");
      die();
}
```

In cui "/Bike_sharing/noleggi.php" corrisponde al nome della pagina, in questo caso quella dei noleggi. Il codice provvederà a verificare che non si stia accedendo al file direttamente, ma solo attraverso "index.php". In caso contrario, il codice reindirizzerà all'index.

Per includere le altre pagine, la pagina index controlla i valori inseriti nell'URL, quindi nell'array GET:

```
if($_GET['page']!=""){
      include($_GET['page'].".php");
}
else if($_GET['page']==""){
      include('home.php');
}
```

Similmente fa per la scelta della lingua da includere:

```
if($_GET['lan']=="" || $_GET['lan']=="ita"){
      include('italiano.php');
}
else if($_GET['lan']=="eng"){
      include('inglese.php');
}
else if($_GET['lan']=="ted"){
      include('tedesco.php');
}
```

Dovendo, per legge, mostrare il banner che avvisa dell'uso dei cookie, il compito è demandato al file "index.php" in javascript:

```
//crea il banner
function createDiv(){
	var bodytag = document.getElementsByTagName('body')[0];
	var div = document.createElement('div');
	div.setAttribute('id','cookie-law');
	div.setAttribute('style','padding-top:10px;animation-
name:scorrimento;animation-duration:0.4s;position:fixed;z-index:2;background-
color:#ffa916;text-align:center;border-radius:200px;margin-
top:20px;width:98%;border:2px solid red;');
	div.innerHTML = '<p><b>Bike Sharing</b> utilizza i cookie, continuando a usare
il sito proseguiremo ad utilizzarli come scritto in <a href="./privacy-cookies-
policy.php/" rel="nofollow" title="Privacy &amp; Cookies Policy">privacy and cookies
policy</a> <button onclick="removeMe()">chiudi</button></p>';

	bodytag.insertBefore(div,bodytag.firstChild); // lo crea prima del <body>
}

window.onload = function(){
	//imposta il cookie "cookie" a "i" quando viene premuto
	if(!document.cookie.includes("cookie=i")){
		createDiv();
	}
};

//al click del pulsante viene richiamata questa funzione che cancella il banner
window.removeMe = function removeMe(){
	var element = document.getElementById('cookie-law');
	element.parentNode.removeChild(element);
	document.cookie = "cookie=i";
};
```

È inoltre suo compito quello di controllare l'esistenza di cookie e di impostare le variabili di sessione, permettendo all'utente di autenticarsi automaticamente.

```
if(isset($_COOKIE[$cookie_mail]) && isset($_COOKIE[$cookie_password])){
	//creo le variabili di sessione
	$_SESSION['email'] = $_COOKIE[$cookie_mail];
	$_SESSION['passw'] = $_COOKIE[$cookie_password];
	//mi collego per ricavare nome e lingua
	$conn = new PDO("mysql:host=$Db_ip;dbname=$Db_database", $Db_user, $Db_password);
	$stmt = $conn->prepare("SELECT * FROM Utenti WHERE Mail='".$_SESSION['email']."'
AND Password='".$_SESSION['passw']."'");
	$stmt->execute();
	$row = $stmt->fetch(PDO::FETCH_ASSOC);
	//imposto nome e lingua nelle variabili di sessione
	if($row){
		$_SESSION['nome']= $row['Nome'];
		$_SESSION['lang']= $row['lingua'];
	}
}
```

- File di configurazione

Il sito deve connettersi al database per estrapolare i dati necessari al corretto funzionamento. Sarebbe una cattiva pratica quella creare ridondanti segmenti di codice "hardcoded" (con variabili scritte a mano). È stato quindi creato un file "config.php", contenente tutti le necessarie variabili per il collegamento al database e per la lettura e l'impostazione dei cookie.

```php
<?php
	$Db_ip = "localhost";
	$Db_user = "root";
	$Db_password = "password";
	$Db_database = "Noleggio_Bici";

	$cookie_mail = "mail_noleggio";
	$cookie_password = "password_noleggio";
?>
```

- File di trilinguismo

Per gestire il trilinguismo è necessario creare ogni paragrafo nell'HTML come una variabile php esplicitata in un altro file .php . In questo modo, in base a quale file .php viene richiamato, le variabili sono sostituite con il loro valore riportato in italiano, tedesco o inglese.

Sono stati creati quindi 3 file: "italiano.php", "tedesco.php" e "inglese.php". Tutti e tre riportano le stesse variabili, ma esplicitate in lingue diverse. "index.php" si occuperà, in base alla lingua selezionata, di includere il file giusto, facendo in modo che le variabili nell'HTML siano sostituite dal loro valore.

italiano.php	**tedesco.php**	**inglese.php**
$primo_titolo = "NOLEGGIO"; $secondo_titolo = "COSTI"; $terzo_titolo = "Orario";	$primo_titolo = "VERMIETUNGEN"; $secondo_titolo = "KOSTEN"; $terzo_titolo = "Zeitplan";	$primo_titolo = "RENTALS"; $secondo_titolo = "COSTS"; $terzo_titolo = "Time table";

- File di header

Il file "header.php" è automaticamente richiamato in ogni pagina. Contiene il menù superiore che mostra le varie pagine accessibili nel sito e deve permettere all'utente di navigare all'interno dello stesso, contenendo link funzionanti.

Per rendere la navigazione intuitiva deve mostrare come evidenziate (cioè verdi) la pagina che si sta visualizzando, modificando quindi l'immagine di sfondo:

```php
//evidenzia il campo "come funziona" nel menu
if($_GET['page']=="come_funziona"){
	echo "<script>document.getElementById('nav_come_funziona').style.background =
\"url('assets/img/autostrada.png')\;
	document.getElementById('nav_come_funziona').style.backgroundRepeat = 'no-repeat';
	document.getElementById('nav_come_funziona').style.backgroundSize = 'cover';</script>";
}
```

Sulla destra notiamo che ci sono 2 pulsanti con le bandierine: sono i pulsanti per il cambio della lingua. "header.php" provvederà, una volta premuto uno dei due, a cambiare la lingua inserendo il campo "lan" nell'URL (creando di fatto una variabile presente nell'array $_GET[]) e a cambiare le bandierine. "index.php" provvederà in base alla lingua selezionata a modificare l'inclusione del file delle lingue (di cui sopra).

```
//Se la pagina è in italiano imposta le bandiere del Tedesco e inglese
if($_SESSION['lang']=="ita" || !isset($_SESSION['lang'])){
	echo "<li class='nav-item' role='presentation'>
			<form method='post' action=''>
				<input type='submit' name='eng' class='nav-link text-
				center text-light' id='eng' style='margin-left:
				10px;width: 15px;height: 30px;background-image:
				url(&quot;assets/img/eng.svg&quot;);background-size:
				cover;background-repeat: no-repeat;background-
				color:white' value=''>
			</form>
		</li>";
	echo "<li class='nav-item' role='presentation'>
			<form method='post' action=''>
				<input name='ted' type='submit' class='nav-link text-
				center text-light' id='ted' style='margin-left:
				5px;width: 15px;height: 30px;background-image:
				url(&quot;assets/img/german.svg&quot;);background-size:
				cover;background-repeat: no-repeat;background-
				color:white' value=''>
			</form>
		</li>";
}
```

L'header provvede inoltre ad aggiornare la lingua per ogni utente, facendo un UPDATE sul database

```
if(isset($_SESSION['email'])){ //se al cambio della lingua è stato eseguito il login
	include('config.php');

	$conn = new PDO("mysql:host=$Db_ip;dbname=$Db_database", $Db_user, $Db_password);
	$mail = $_SESSION['email'];
	//imposta la lingua in italiano
	$stmt = $conn->prepare("UPDATE Utenti SET lingua='ita' WHERE Mail='$mail'");
	$stmt->execute();
}
```

"header.php" deve inoltre provvedere a inserire un nuovo link una volta loggato un utente. Modifica quindi i link "Login" e "Register" nel link alla pagina dell'utente (mostrando il suo nome) e il link "Logout".

- File di policy

Come obbliga la legge, è stata creata una pagina dedicata alla policy dei cookie. La pagina è molto semplice: mostra solo un testo che specifica l'intento didattico del sito e l'uso dei cookie al solo fine della memorizzazione dei dati d'accesso, quindi per la semplificazione dell'uso del sito all'utente.

Cookie e Privacy

Il sito Bike Sharing usa i cookie per semplificare la navigazione nel sito mantenendo i dati di accesso dell'utente in modo da non essere obbligato a rifornirli.

L'uso dei cookie avviene solo quando l'utente lo specifica al login

- File per le timbrature

Viene inserito anche un file php accessibile solo dal gestore, accedendo con l'account "admin". Mostra tutte le timbrature aggiornandosi in automatico. È un esempio di come il gestore possa tenere sotto controllo i noleggi in tempo reale.

Interfaccia da ADMIN - Logout

Nome	Cognome	Mail	Data inizio prestito	Indirizzo di inizio prestito	Data di consegna	Indirizzo di fine consegna	Costo noleggio	Pagato
Carlo	Gabardi	carletto@gmail.it	2019-04-01 00:00:00	Piazza Verdi	2019-04-02 00:00:00	Piazza Verdi	1€	No
Carlo	Gabardi	carletto@gmail.it	2021-04-04 00:00:00	Piazza Verdi	2019-04-01 00:00:00	Piazza Verdi	1€	No
Carlo	Gabardi	carletto@gmail.it	2020-04-10 00:00:00	Piazza Municipio	2020-04-30 00:00:00	Piazza Bruni	888€	Si

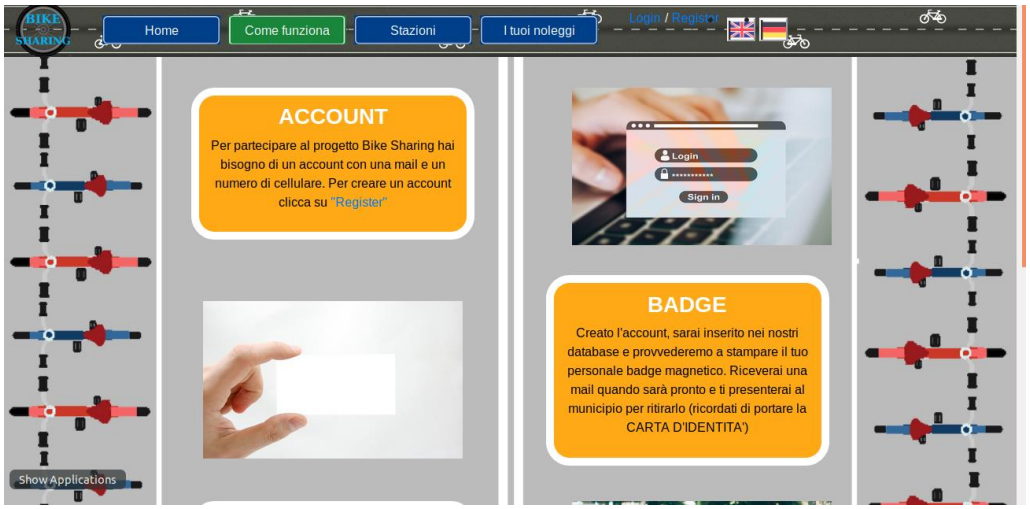
Pagine Web

- Home

La pagina Home è la prima che si visualizza all'ingresso nel sito, mostra quindi una piccola presentazione del servizio con due paragrafi e due immagini. L'incolonnamento dei paragrafi e delle immagini è uguale in tutte le pagine, ritenendo che fosse il più semplice e funzionale.

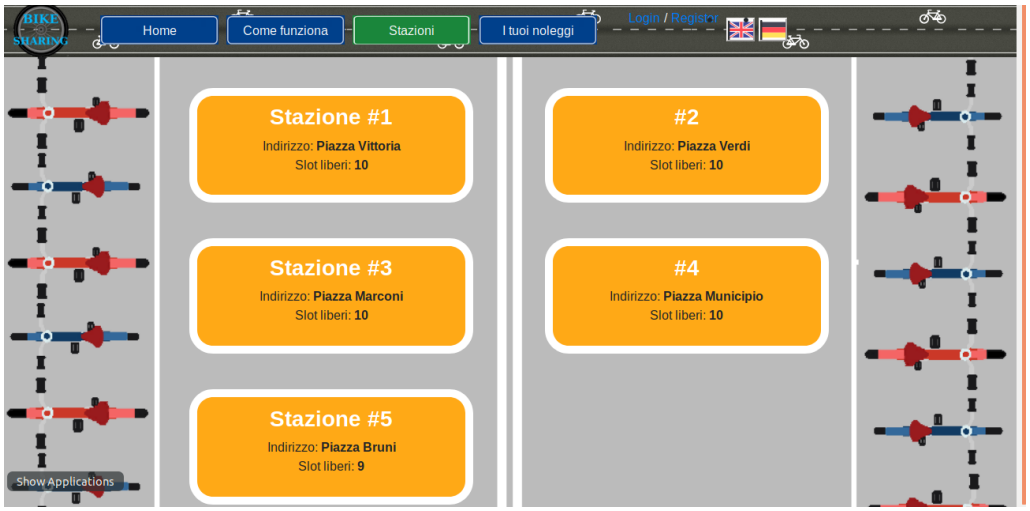
- Sezione "come funziona"

La sezione "Come funziona" spiega con pochi paragrafi ed immagini come usufruire del servizio, quindi come registrarsi e come usare il badge nelle stazioni di noleggio.



- Sezione “stazioni”

La sezione “Stazioni” contiene la lista di tutte le stazioni di noleggio (in ordine di slot liberi), con relativo indirizzo e numero di slot liberi.



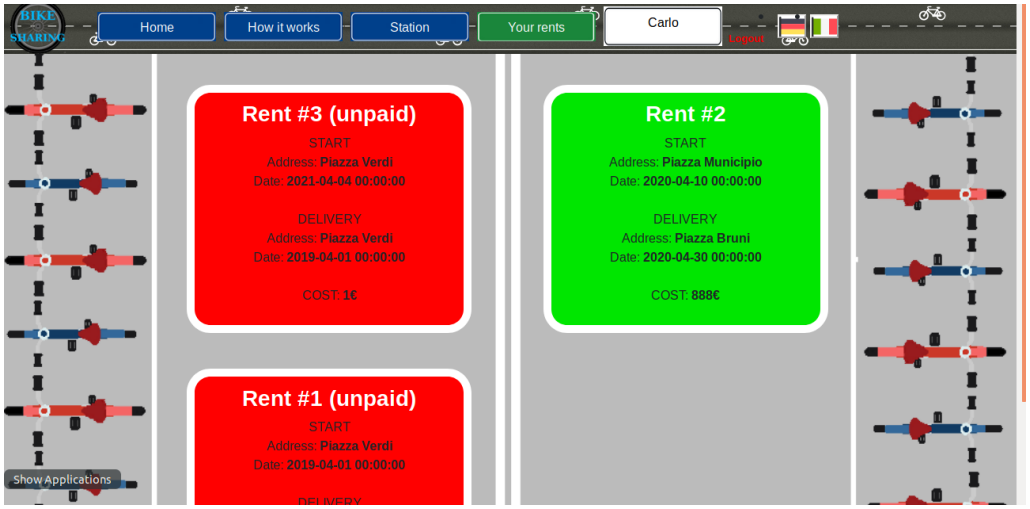
Anche in questa pagina viene interrogato il database con una SELECT piuttosto semplice:

```
SELECT * FROM Stazioni ORDER BY Slot_liberi
```

- Sezione “noleggi”

La sezione “Noleggi” mostra i noleggi dell’utente se è loggato, altrimenti mostra un messaggio di invito all’autenticazione.

Al login, la pagina mostra tutti i noleggi con i relativi dati e il costo. Se sono già stati pagati saranno con sfondo verde, altrimenti saranno di colore rosso. La detrazione del costo dal saldo di ogni utente è fatta in automatico dal lettore RFID (se il saldo è sufficiente).

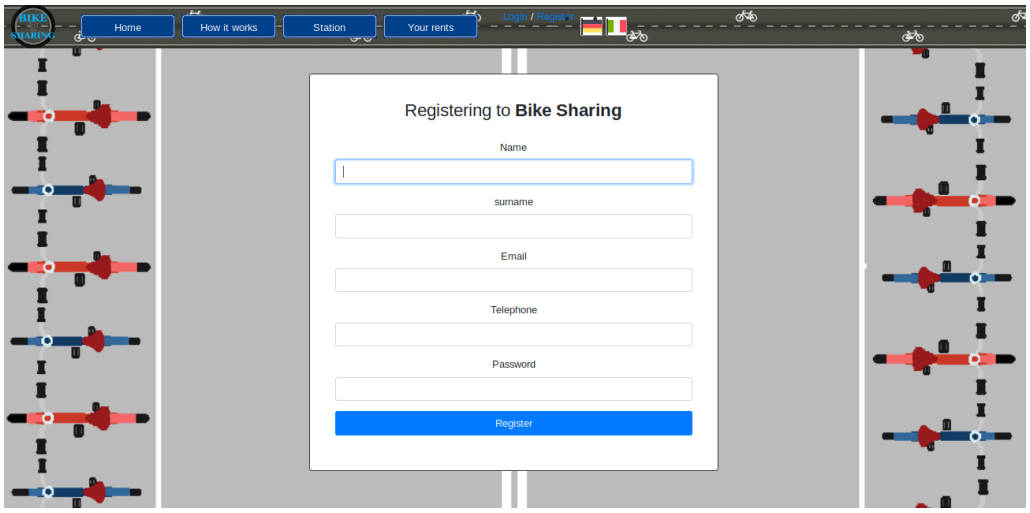


La query di SELECT è più complicata, dovendo mostrare i dati in comune tra la tabella Utenti, Stazioni e Noleggia in ordine di data (in modo che il più recente sia mostrato prima).

```
SELECT Data_prestito, Data_consegna, Costo, Pagato, S1.Indirizzo
AS partenza, S2.Indirizzo AS arrivo
FROM Utenti, Noleggia, Stazioni AS S1, Stazioni AS S2
WHERE Utenti.Mail = '".$_SESSION['email']."'
AND Utenti.Id_utente = Noleggia.Id_utente
AND Costo!='NULL'
AND Noleggia.Stazione_partenza = S1.Id_stazione
AND Noleggia.Stazione_arrivo = S2.Id_stazione
ORDER BY Data_prestito DESC
```

- Register

La pagina di registrazione è un semplice form di immissione dati, in cui l'utente deve specificare il suo nome, cognome, mail, telefono e password.



In questa pagina viene eseguito un INSERT dell'utente nel database, in cui si impone il suo ID a 0 (il database provvederà a inserirne uno valido, essendo il campo un AUTOINCREMENT) e il "Saldo" a 0

```
INSERT INTO Utenti
VALUE ('0', '$u_nome', '$u_surname', '$u_password', '$u_mail', '$u_cell', '0', 'ita')
```

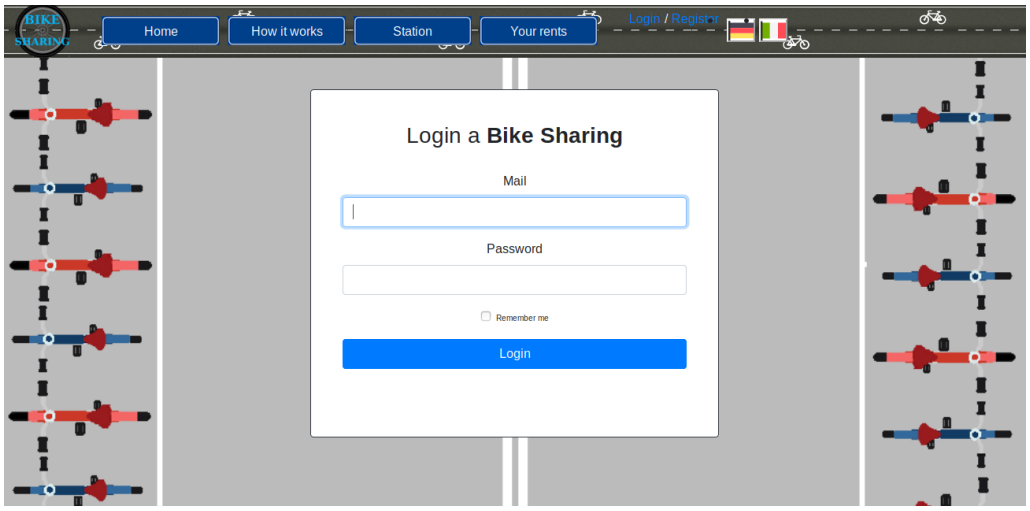
In questa pagina avviene la criptazione della password a lato browser, sfruttando una funzione javascript del file “sha512.js” che implementa la crittografia sha512 appunto:

```
document.getElementById("pass").addEventListener("input", Cripto);
function Cripto(){
	var password = document.getElementById("pass").value;
	document.getElementById("pass-cripto").value = hex_sha512(password);
}
```

La versione criptata della password viene mantenuta in un campo di testo nascosto (hidden) dalla quale viene poi estratta per il controllo sul database. Questo tipo di crittografia quindi non rende il sito sicuro, permettendo a degli hacker di fare MITM (Man in the middle) e di ricavare la password salvata sul database.

- Login

La pagina di login è costituita da un semplice form per l’immissione dei dati con un riquadro per abilitare i cookie. La pagina provvede a fare il login e a impostare i cookie, se viene esplicitato dall’utente.



La query è molto semplice:

```
SELECT * FROM Utenti WHERE Mail='$mail' AND Password='$password'
```

Se è portata a termina, rimanda l'utente alla home, impostando anche le variabili di sessione, di modo che l'header visualizzi il link all'account e al logout.

```
if(isset($_POST['mail'])){
        $mail = $_POST['mail'];
        $password = $_POST['pass-cript'];

        $conn = new PDO("mysql:host=$Db_ip;dbname=$Db_database", $Db_user, $Db_password);
        $stmt = $conn->prepare("SELECT * FROM Utenti WHERE Mail='$mail' AND Password='$password'");
        $stmt->execute();
        $row = $stmt->fetch(PDO::FETCH_ASSOC);

        if($row){
                //crea variabili di sessione
                $_SESSION['email'] = $mail;
                $_SESSION['passw'] = $password;
                $_SESSION['nome']= $row['Nome'];
                $_SESSION['lang']= $row['lingua'];

                //se l'utente vuole I cookies
                if(isset($_POST['cookie'])){
                        //li mette in javascript perchè in PHP non è possibile da manuale
                        echo "<script>document.cookie = '".$cookie_mail."=".$mail."; expires=Thu, 18
                Dec 2220 12:00:00 UTC';</script>";
                        echo "<script>document.cookie = '".$cookie_password."=".$password.";
expires=Thu, 18 Dec 2220 12:00:00 UTC';</script>";
                }

                //manda alla home
                echo "<script type='text/javascript'>location.href
                        ='index.php?lan=".$_SESSION['lang']."';</script>";
        }
        else{
                echo "<br><h4 align=center style='color:red'>Login fallito!!!</h4>";
        }
}
```

- Sezione "user"

La sezione dedicata all'utente mostra semplicemente la sua mail, il suo nome, il numero di telefono, il numero di noleggi e il saldo.

La query è semplice, dovendo selezionare tutti i campi della tabella utenti per l'utente registrato.

```
SELECT * FROM Utenti WHERE Mail='$mail' AND Password='$password'
```

È inoltre presente la query per il contro dei noleggi:

```
SELECT COUNT(*) as num FROM Utenti, Noleggia
WHERE Utenti.Mail = '$mail' AND Utenti.Id_utente = Noleggia.Id_utente
```

Eseguita la query, si creano delle variabili per contenere i valori da mostrare nel riquadro al centro:

```
$mail = $_SESSION['email']; $password = $_SESSION['passw'];

$conn = new PDO("mysql:host=$Db_ip;dbname=$Db_database", $Db_user, $Db_password);
$stmt = $conn->prepare("SELECT * FROM Utenti WHERE Mail='$mail' AND
Password='$password'");
$stmt->execute();
$row = $stmt->fetch(PDO::FETCH_ASSOC);
//usate nel <div> in HTML
$_SESSION['cell'] = $row['Cellulare'];
$_SESSION['saldo'] = $row['Saldo'];
$_SESSION['nome'] = $row['Nome'];
$_SESSION['cognome'] = $row['Cognome'];

$stmt = $conn->prepare("SELECT COUNT(*) as num FROM Utenti, Noleggia WHERE Utenti.Mail
= '$mail' AND Utenti.Id_utente = Noleggia.Id_utente");
$stmt->execute();
$row = $stmt->fetch(PDO::FETCH_ASSOC);
$_SESSION['noleggi'] = $row['num']; //usata nel <div>
```

- Logout

La pagina "logout.php" provvede semplicemente a eliminare i cookies e le variabili di sessione e a tornare alla home.

```
echo "<script>document.cookie = '".$cookie_mail."=".$_SESSION['email']."; expires=Thu,
18 Dec 2000 12:00:00 UTC';</script>";
echo "<script>document.cookie = '".$cookie_password."=".$_SESSION['passw']."';
expires=Thu, 18 Dec 2000 12:00:00 UTC';</script>";
unset($_SESSION['email']);
unset($_SESSION['passw']);
echo "<script type='text/javascript'>location.href =
'index.php?lan=".$_SESSION['lang']."';</script>";
```

INSTALLAZIONE DEL SERVER WEB E DEL DATABASE

Prima di dedicarsi alla creazione del sito o del simulatore della stazione, è stato necessario installare i componenti del server (eseguendo il login come root):

Apache

Per installare Apache2 su una macchina con sistema operativo debian è sufficiente digitare il comando

```
apt-get install apache2
```

Si creano così i file di configurazione in "`/etc/apache2`" e si crea la directory "`/var/www/html`" dove vengono salvati i file del sito.

PHP7.0

Per installare PHP7.0 è sufficiente digitare il comando

```
apt-get install php7.0-cli
```

Per permettere la comunicazione tra apache2 e php7.0 è necessario battere il comando

```
apt-get install libapache2-mod-php7.0
```

MySQL

È necessario poi installare il database con il comando

```
apt-get install mysql-server php7.0-mysql
```

PhpMyAdmin

Per installare PhpMyAdmin basta inserire il comando

```
apt-get install phpmyadmin
```

REALIZZAZIONE SIMULATORE DI STAZIONE

INTRODUZIONE

Per funzionare, i due raspberry sono programmati per collegarsi in automatico ad una rete wifi senza password, cosicché siano nella stessa rete e possano comunicare tra di loro. Collegandosi all'hotspot di uno stesso cellulare, ai raspberry vengono dati sempre gli stessi ip, consentendoci di impostare "hardcoded" i collegamenti tra i due (specificando cioè un indirizzo ip).

Inoltre, al simulatore della stazione è stato programmato di far partire in automatico il software in python per la gestione dei noleggi. In questo modo ci basta collegare i due raspberry all'alimentazione e accendere l'hotspot wifi del cellulare per attivare tutti i servizi.

Per collegarci al database in python usiamo l'API PEP249 che permette, attraverso delle funzioni prestabilite, di comunicare con il database MySQL e di eseguire varie query. L'API usa il protocollo TCP/IP, con la possibilità di appoggiarsi al SSL per rendere sicura la comunicazione. Usandolo a scopo didattico, ci siamo appoggiati alla API senza rendere sicura la connessione.

ACCENNI TEORICI SUL RFID

In telecomunicazioni ed elettronica con l'acronimo RFID (Radio-Frequency IDentification) si intende una tecnologia per l'identificazione di informazioni basata sulla capacità di memorizzazione di dati da parte di particolari tessere magnetiche, chiamate tag ("badge" nel progetto), e sulla capacità di queste di rispondere all'interrogazione a distanza da parte di appositi apparati fissi o portatili, chiamati reader.

Questa identificazione avviene mediante radiofrequenza, grazie alla quale un reader è in grado di comunicare e aggiornare le informazioni contenute nei tag che sta interrogando (nonostante il nome "reader" cioè lettore, è in grado anche di scrivere).

L'elemento che caratterizza un sistema RFID è il tag (passivo nel nostro caso): è un dispositivo elettronico composto da un chip ed un'antenna a radio frequenza montati su un "substrato" che ha anche il compito di sostenerli. Il chip (grande pochi millimetri) è la parte "intelligente" costituita da una memoria non volatile ed un codice in genere univoco (UID), il quale viene trasmesso tramite l'antenna RF (la spira risonante o circuito di trasmissione del segnale wireless) all'apparato lettore che leggerà i dati ricevuti o li aggiornerà.

Il lettore emette un campo elettromagnetico che tramite induzione genera nell'antenna del tag una corrente che alimenta il chip. Il chip così alimentato comunica le sue informazioni che vengono irradiate tramite l'antenna al lettore.

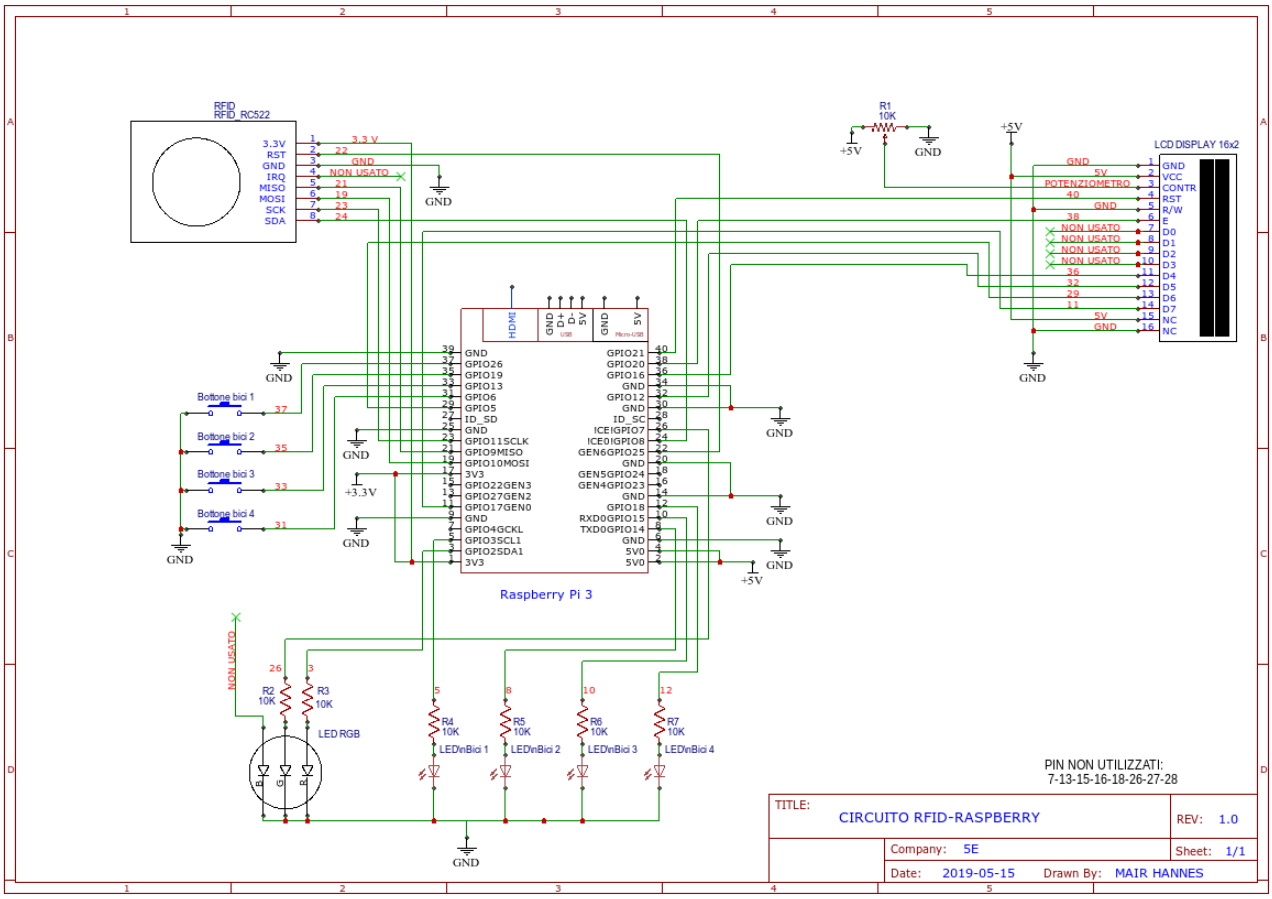
La tecnologia RFID ha diversi vantaggi rispetto alle tradizionali tecnologie dei codici a barre e delle bande magnetiche: è contactless, quindi non necessita di venire a contatto col lettore; non deve essere visibile (come ad esempio i codici a barre) e la comunicazione è molto veloce e può essere cifrata (nel nostro caso è in chiaro)

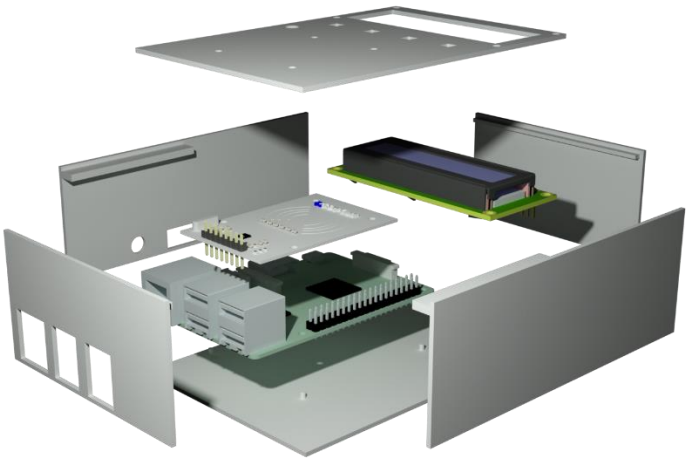
Esempi quotidiani di RFID sono l'ABO negli autobus, che tramite un lettore costituiscono un sistema praticamente uguale al nostro, e il tesserino della macchinetta del caffè, il quale comunica con la macchinetta in modo cifrato.

Lettore RFID

ASSEMBLAGGIO DEL SIMULATORE

Il circuito del simulatore installato sul raspberry comprende un LCD, 5 LED, 4 pulsanti e un lettore RFID. È stato realizzato con EasyEDA, un software che ci ha permesso di creare il disegno circuitale in modo semplice e funzionale:





I pezzi del simulatore

- LCD

Il LCD da 16x2 segmenti permette all'utente di visualizzare semplici messaggi per orientarsi nel funzionamento del simulatore come "Avvicinare il badge", "Scegliere la bici", "Utente registrato". Mostra inoltre la conferma della bici scelta e il costo del noleggio alla consegna della bici.

- LED

Sono installati complessivamente 5 LED: 4 per simulare le bici e 1 per uso generico.

Il LED ad uso generico è utilizzato per confermare l'autenticazione dell'utente e per confermare l'inizio del noleggio accendendosi di verde; si accende invece di rosso quando l'utente non è riconosciuto.

I 4 led sono stati installati per far capire all'utente quali delle 4 bici è disponibile. Alla selezione della bici i LED accesi indicheranno quelle disponibili per il noleggio (quelle che in una stazione reale sono presenti e legate ad un lucchetto elettronico) mentre quelli spenti indicheranno che la bici è già stata noleggiata. Sono posti esattamente sopra ai pulsanti, in modo che ogni pulsante si riferisca chiaramente ad una bici, cioè ad un LED.

- Pulsanti

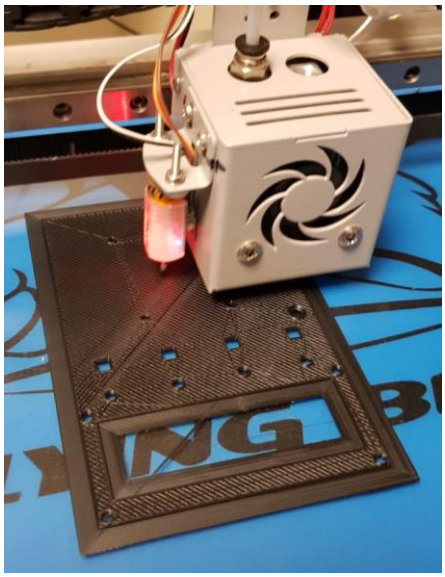
Non potendo usare delle vere bici, sono stati installati 4 pulsanti sul raspberry. In questo modo l'utente, pigiando uno dei 4 pulsanti, potrà scegliere la bici che vorrà usare nel noleggio.

- Lettore

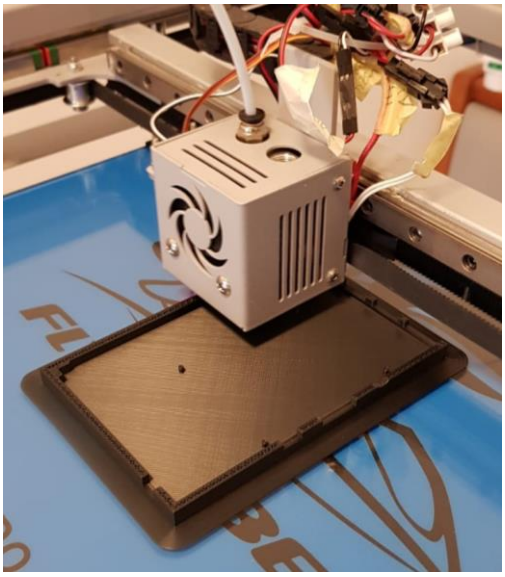
Il lettore RFID è utilizzato solo per leggere il codice univoco dei badge, codice usato come chiave primaria (quindi come identificatore dell'utente) nella tabella "Utenti". Non viene scritto nulla nei vari badge, limitandosi il lettore a leggere il codice UID inviatogli dai tag.

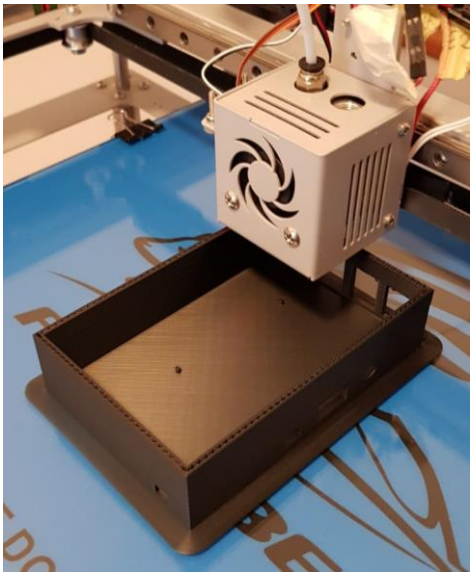
- Involucro di plastica

Il rivestimento di plastica nera è stato inizialmente progettato in AutoCAD, il quale, dopo aver preso le adeguate misure, ci ha permesso di progettare al millimetro un contenitore in grado di contenere il cablaggio e i componenti hardware. È stata ideata in 2 parti: il coperchio e il resto del contenitore; questo per evitare di creare piccoli pezzi che potessero rompersi facilmente. Dopo averla esportata in STL è stata inserita all'interno del software della stampante 3D ed è stata stampata in 11 ore.




MARCOSTASE
RFID






MARCOSTASE
RFID

FASI DI FUNZIONAMENTO

Riassumiamo le fasi di funzionamento del simulatore, ricordando che all'accensione il raspberry attende mostrando la scritta "Avvicinare badge":

- Lettura del badge

All'avvicinamento di un badge sul lettore RFID il raspberry farà un controllo sul database per verificare se l'utente è registrato o meno. Se è registrato mostra la scritta "Utente registrato" sul LCD e accende di verde il LED generico; se l'utente non è registrato mostra la scritta "Utente non registrato" e accende il LED di rosso.

- Proposta delle bici

All'utente che non ha un noleggio in corso vengono proposte le bici disponibili per un noleggio. Il raspberry fa un controllo sul database per verificare la disponibilità delle bici e accende di conseguenza i 4 LED. Successivamente invita l'utente a scegliere una bici con la scritta "Scegliere bici".

L'utente può premere il pulsante corrispondente alla bici che vuole noleggiare.

- Inizio noleggio

Alla pressione di uno dei pulsanti, il raspberry "aprirà" il lucchetto iniziando il noleggio. Esegue quindi una query nella tabella "Noleggia" con i relativi dati utente e della bici e mostra sull'LCD la conferma della bici scelta. Torna quindi nell'assetto di partenza.

- Termine noleggio

Quando un utente avvicina il badge al lettore, il raspberry, dopo aver controllato se l'utente esiste nel database, controlla se ha già iniziato un noleggio. Se lo ha già iniziato è costretto a riconsegnare la bici prima di iniziarne un altro.

Mostra quindi sul LCD "Bici x riconsegnata", mostrando il numero della bici, e calcola automaticamente il costo del noleggio in base al tempo di utilizzo e in base al costo orario e lo mostra sul LCD. Interrogando inoltre il database, verifica che il saldo sia sufficiente a pagare il costo e, se effettivamente è sufficiente, lo paga in automatico.

ACCENNI ALLA PROGRAMMAZIONE PYTHON

Il python è il linguaggio usato nel raspberry che simula il lettore delle stazioni di noleggio. Come linguaggio si occupa di sincronizzare le varie funzionalità del raspberry: l'accensione dei led, la lettura dei pulsanti, il collegamento al database e la gestione del LCD.

Il programma si occupa di accendere diverse volte i vari LED. Per fare questo il comando mette come "HIGH" il pin interessato:

```
GPIO.output(7,GPIO.HIGH)
```

La gestione del LCD è anch'essa molto semplice, dovendo semplicemente "pulirlo" con un clear(), per poi immettere il messaggio con semplicità:

```
lcd.clear()
lcd.message("     UTENTE     \n   REGISTRATO   ")
```

Infine la connessione al database è molto frequente per inserire e leggere dati. Il codice che se ne occupa è più complesso dei precedenti, ma lavora similmente al PDO o al MySQLi:

```
mydb = mysql.connector.connect(host=ipserver,
port="3306",user="admin",passwd="admin",database="Noleggio_Bici")
cursor = mydb.cursor()
query_select = "SELECT COUNT(*) FROM Noleggia WHERE Id_utente='"+codice+"' AND Costo=0"
cursor.execute(query_select)
```

ANALISI DEL CODICE NELLE FASI DI FUNZIONAMENTO

Si passa in rassegna ogni operazione svolta dal simulatore di lettore della stazione di noleggio:

Lettura del badge e controllo nel database

Appena passato il badge sul lettore RFID, il simulatore va a verificare che il codice estrapolato esista nel database e sia associato a un utente.

```
lcd.message("AVVICINARE BADGE")

id,text = reader.read()
codice = str(id)    #change id (int) into string

query_select = "SELECT COUNT(*) FROM Utenti WHERE Id_utente='"+codice+"'"
cursor.execute(query_select)
result = cursor.fetchmany(1)
for row in result:
      numero = (row[0])   #numero=1 : exist || numero=0 : not exists
```

Controllo delle bici nel database

Una volta verificato l'utente, controlla nel database quale bici è in uso e quale no. In questo modo proporrà all'utente solo quelle disponibili accendendo il relativo LED.

```
cursor.execute("SELECT Id_bici FROM Bici WHERE In_uso='0'")
result = cursor.fetchall()
      #Se la bici e disponibile la imposto a 1
for row in result:
      if row[0] == 1:
            led_bici1.on()
            bici1 = 1
      if row[0] == 2:
            led_bici2.on()
            bici2 = 1
      if row[0] == 3:
            led_bici3.on()
            bici3 = 1
      if row[0] == 4:
            led_bici4.on()
            bici4 = 1
```

Inizio noleggio e uso della bici

Scelta la bici, viene creata una entry nella tabella "Noleggi", inserendo i relativi dati utente, bici e di stazione. Inoltre va ad impostare come "In uso" la bici scelta dall'utente.

```
query_aggbici = "UPDATE `Bici` SET `In_uso`='1' WHERE `Bici`.`Id_bici`='"+numerobici+"'"
cursor.execute(query_aggbici)
mydb.commit()
```

```
query_noleggia = "INSERT INTO `Noleggia`
(`Id_utente`,`Id_bici`,`Data_prestito`,`Data_consegna`,`Stazione_partenza`,
`Stazione_arrivo`,`Costo`,`Pagato`)
VALUES('"+codice+"','"+numerobici+"',CURRENT_TIMESTAMP,NULL,'"+idstazione+"',NULL,'0',NULL)"
cursor.execute(query_noleggia)
mydb.commit()
```

Termine del noleggio

Una volta che un utente passa il badge per terminare un noleggio, questo viene terminato. Subito dopo la verifica dell'utente, viene verificato se nella tabella "Noleggi" c'è un noleggio in corso. Se non c'è, continua proponendo all'utente quale bici vuole usare; se è presente un noleggio in corso, lo termina, ipotizzando che per ogni badge può essere noleggiata una sola bici alla volta.

Viene quindi impostata come disponibile la bici e viene terminato il noleggio, inserendo la data di fine noleggio. Viene inoltre calcolato il costo in base alla fascia oraria e, se il saldo è sufficiente, il noleggio viene pagato automaticamente.

```
query_completamento = "UPDATE`Noleggia`SET`Data_consegna`=CURRENT_TIMESTAMP,
`Stazione_arrivo`='"+idstazione+"',`Costo`='0'WHERE`Noleggia`.`Id_utente`='"+codice+"'
AND`Costo`=0"
cursor.execute(query_completamento)
mydb.commit()
```

```
if(saldo>costo):
      query_saldo = "UPDATE `Utenti` SET `Saldo`='"+str(round(saldo,2)-
round(costo,2))+"' WHERE `Id_utente`='"+codice+"'"
      cursor.execute(query_saldo)
      mydb.commit()
      query_noleggio ="UPDATE `Noleggia` SET `Pagato`='1' WHERE `Id_utente`='"+codice+"'
ORDER BY `Data_prestito` DESC LIMIT 1"
      cursor.execute(query_noleggio)
      mydb.commit()
```

RIFERIMENTI E FONTI

Berardi, M. D., s.d. *Prove scritte Sistemi e Informatica Esame di Stato.* [Online]
Available at:
http://lnx.maurodeberardis.it/index.php?option=com_jdownloads&Itemid=191&view=viewcategory&catid=18

Bragadin, F., s.d. *Whymatematica.* [Online]
Available at: http://www.whymatematica.com/

Debian Development Team, s.d. *Package: php7.0 (7.0.33-0+deb9u3).* [Online]
Available at: https://packages.debian.org/stretch/php7.0

draw.io Inc, s.d. *Draw.io.* [Online]
Available at: https://www.draw.io/

Free stock Inc., s.d. *Free stock photos.* [Online]
Available at: https://www.pexels.com/

Marzocchella, A., s.d. *Modello sito passo passo 2918-2019.* [Online]
Available at: https://docs.google.com/document/d/1s4s1TFhpXTmfqnR1wQmh43FEfuea1llEc-5WuzyCnqI/edit

MySQL Development Team, s.d. *Documentation.* [Online]
Available at: https://dev.mysql.com/doc/

Oracle Corporation, s.d. *Introduction to MySQL Connector/Python.* [Online]
Available at: https://dev.mysql.com/doc/connector-python/en/connector-python-introduction.html

phpMyAdmin developers, s.d. *Documentation.* [Online]
Available at: https://www.phpmyadmin.net/

Ubuntu development Team, s.d. *HTTPD - Server web Apache2.* [Online]
Available at: https://help.ubuntu.com/lts/serverguide/httpd.html

Unsplash Inc., Unsplash. *Photos for everyone.* [Online]
Available at: https://unsplash.com/

Wikipedia Foundation, s.d. *Radio-frequency identification.* [Online]
Available at: https://it.wikipedia.org/wiki/Radio-frequency_identification