

Delfrate Riccardo

**5B Informatica e telecomunicazione articolazione informatica**

## **Relazione Tecnica**

|  |  |
| --- | --- |
| La seguente relazione illustrerà il funzionamento, le procedure e gli strumenti utilizzati nel mio elaborato.  **INDICE**  **Richieste:**  Un’azienda di e-Commerce richiede la realizzazione di un sito dinamico che permetta la navigazione del sito senza alcun tipo di registrazione.  Si preveda la registrazione, attraverso l’inserimento di mail e di password, per i clienti che vorranno effettuare degli acquisti.  Il sito dovrà prevedere la gestione del carrello.  Dopo aver effettuato il login il cliente dovrà aver inserito nel carrello almeno un elemento in modo tale che, per gli accessi successivi, ritroverà le scelte precedentemente effettuate pronte per un eventuale acquisto, che avverrà con una carta di credito con una modalità protetta.  Il candidato, con riferimento a quanto descritto, facendo le opportune ipotesi aggiuntive, produca un elaborato che:  **Sistemi**   * La metodologia e i protocolli utilizzati per lo scambio delle informazioni sulla carta di credito in modo criptato tramite algoritmo RSA. * Approfondisca anche in ambito matematico l’algoritmo RSA.   **Informatica**   * Realizzi il progetto della base di dati per la gestione di quanto descritto, in particolare si richiedono: il modello concettuale, il corrispondente modello logico e fisico. * Disegni e realizzi un sito web che consenta la realizzazione delle funzionalità sopra descritte. | |
| **Progettazione**  **Requisiti funzionali:**   * Un utente per poter procedere all’acquisto dovrà essere registrato. * Saranno presenti 2 barre poste una lateralmente e l’altra sovrastante la visualizzazione dei prodotti che permetteranno di raggiunge le funzionalità del sito. * Sarà presente un form di registrazione per permettere di registrarsi. * Ogni utente una volta registrato avrà la possibilità di accedere alla sua area riservata. * L’area riservata conterrà un carrello con il riepilogo di tutti i prodotti selezionati. * La parte al di fuori dell’area riservata sarà visibile a tutti. * Sarà presente un form per l’inserimento dei dati sulla carta di credito. * Sarà presente un form per l’inserimento dei dati di spedizione. * Sarà presente un bottone di logout per effettuare la disconnessione.   **Requisiti non funzionali:**   * In fase di registrazione sarà presente un controllo per verificare che le due password inserite corrispondano. * Prima di avanzare alla fase di pagamento sarà obbligatorio accettare i termini e le condizioni. * Una volta effettuato l’accesso verranno visualizzati bottoni per utilizzare funzionalità extra. * In caso di eventuali errori/operazioni avvenute verranno comunicati/e all’utente tramite alert. * Tutti i dati dell’utente che ha effettuato la registrazione saranno salvati nel database. * La password dovrà necessariamente essere di almeno 8 caratteri e contenere almeno un numero, almeno una lettera maiuscola e almeno una lettera minuscola. * La password prima di venire inserita verrà criptata utilizzando due metodologie hash differenti. * Ogni articolo apparterrà ad una categoria.   **Work Breakdown Structure:** | |
|  | |
| **Diagramma di Gantt:** | |

* **Pianificazione:**

Pianificazioni di tutte le varie attività successive, brainstorming, valutazioni delle opzioni, valutazione dei rischi, studio fattibilità del progetto, consulto fonti.

* **Progettazione:**

Realizzazione analisi del problema, realizzazione schema ER, realizzazione schema logico.

* **Layout:**

Realizzazione del layout del sito, scelta dei colori, del font e del template.

* **Database:**

Realizzazione schema ER schema logico e creazione fisica del database in Mysql.

Identificazione delle entità e delle relazioni.

* **Login:**

Creazione del form di login per permettere all'utente di accedere all'area riservata.

* **Registrazione:**

Creazione del form di registrazione che permetterà di essere aggiunti nel database con relative operazioni per garantirne la sicurezza.

* **Logout:**

Implementazione pagina di logout la quale permetterà di disconnettersi chiudendo la sessione ed eliminando tutti i dati contenuti in essa.

* **Aree-Riservate:**

Implementazione aree riservata per amministratore e utente standard.

L'amministratore avrà una panoramica di tutte le operazioni che potranno essere eseguite nel database e visualizzare un riepilogo dei prodotti, le categorie e gli utenti presenti.

L'utente standard una volta avvenuto il login verrà riportato nella home page dove compariranno il bottone per la visualizzazione del carrello il quale potrà essere usato per il completamento dell'ordine e il bottone di logout.

* **Prodotti:**

Realizzazione della parte di visualizzazione dei prodotti presenti nel database.

* **Pagamento:**

Implementazione form di inserimento dei dati relativi alla carta di credito e implementazione dell'algoritmo RSA per la sicurezza di essi.

* **Spedizione:**

Realizzazione form per l'inserimento dei dati relativi all'indirizzo di spedizione e di fatturazione.

* **Test:**

Verifica di tutte le funzionalità offerte e individuazione di eventuali errori.

* **Risoluzione problemi:**

Risoluzione dei problemi emersi dalla fase di test.

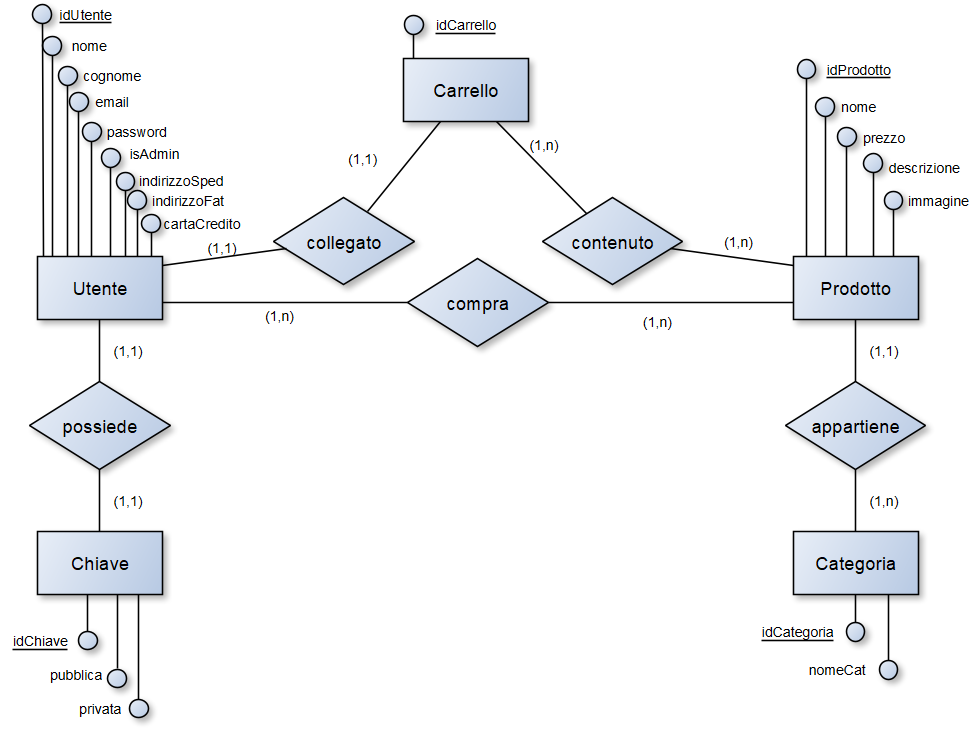
* **Consegna:**

Consegna finale progetto.

**Ipotesi aggiuntive:**

* Amministratore può effettuare operazioni di inserimento.
* Per fare acquisti bisogna registrarsi in modo da essere inseriti in un database.
* La password verrà criptata in modo adeguato prima di essere inserita nel database
* Sarà presente un campo nella tabella utente per distinguere l’amministratore dall’utente standard.
* Sarà presente un carrello contenente tutti i prodotti con costo finale e bottone per procedere all’acquisto.
* Il form fi login sarà unico, con reindirizzamelo in pagine diverse a seconda dei permessi

**Schema ER:**



**Schema logico:**

Chiave: (idChiave(PK), pubblica, privata);

Utente: (idUtente(PK), nome, cognome, email, password, isAdmin, via, cap, nCiv, comune, provincia, FragioneSociale, FpecCod, FpartitaIva, Fvia, Fcap, FnCiv, Fcomune, Fprovincia, FcodFiscale, cartaCredito idChiave(FK), idCarrello(FK));

Compra(idCompra(PK), idUtente(FK), idProdotto(FK));

Prodotto: (idProdotto(PK), nome, prezzo, descrizione, immagine, idCategoria(FK));

Carrello: (idCarrello(PK));

Contenuto(idContenuto(PK), idCarrello(FK), idProdotto(FK));

Categoria: (idCategoria(PK), nomeCat);

**Analisi associazioni:**

**Chiave-Utente:**

Tra l’entità chiave e l’entità utente è presente un’associazione “possiede” uno a uno in quanto un utente possiede a una sola coppia di chiavi e una coppia di chiavi è posseduta da un solo utente.

**Utente-Carrello:**

Tra l’entità Utente e l’entità Carrello è presente un’associazione “collegato” uno a uno in quanto un utente è collegato ad un solo carrello e un carrello è collegato ad un solo utente.

**Utente-Prodotto:**

Tra l’entità Utente e l’entità Prodotto è presente un’associazione “compra” molti a molti in quanto un utente può comprare più prodotti e un prodotto può essere acquistato da più utenti.

**Carrello-Prodotto:**

Tra l’entità Carrello e l’entità Prodotto è presente un’associazione “contenuto” molti a molti in quanto un carrello può contenere più prodotti e un prodotto può essere contenuto in più carrelli.

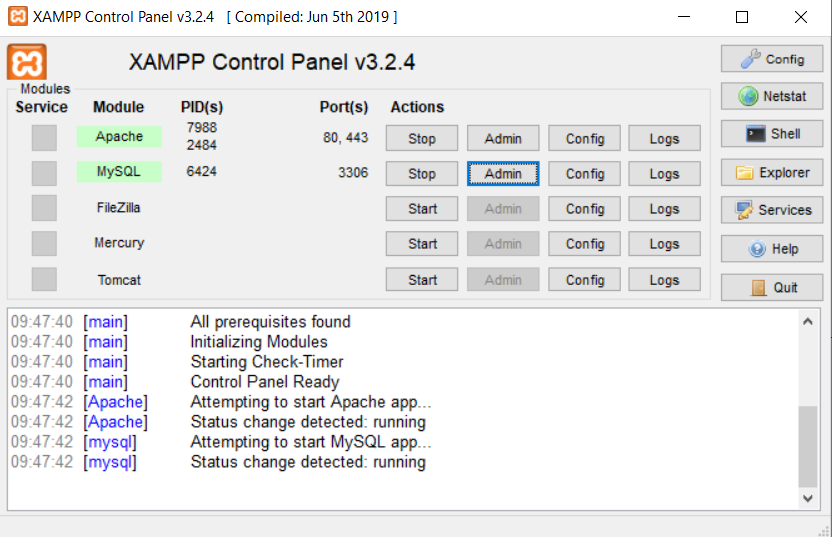
**Prodotto-Categoria:**

Tra l’entità Prodotto e l’entità Categoria è presente un’associazione “appartiene” uno a molti in quanto un prodotto appartiene a una sola categoria e a una categoria appartengono più prodotti.

**Strumenti utilizzati:**

**XAMPP:**

Ho deciso di utilizzare XAMPP in quanto offre la possibilità di gestire un Application server capace di interpretare pagine web dinamiche e un modulo MySQL il quale permette di avere a disposizione un database.

****

**Linguaggi di programmazione utilizzati:**

**PHP:**

PHP è un linguaggio di scripting lato server interpretato, concepito per la programmazione di pagine web dinamiche. L'interprete PHP è un software libero distribuito sotto la PHP License.

Ho deciso di utilizzare questo linguaggio in quanto è in grado di interfacciarsi a innumerevoli DBMS tra cui MySQL per il quale è ottimamente integrato e Fornisce un'API specifica per interagire con il web-server Apache.

**Javascript:**

JavaScript è un linguaggio di programmazione orientato agli oggetti e agli eventi lato client, ho utilizzato questo linguaggio in quanto offre una grande interoperabilità con html e php.

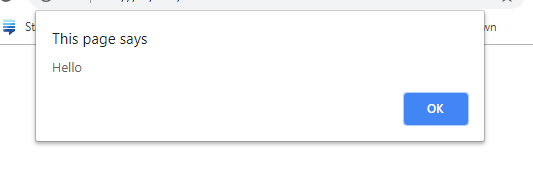
**Sweet alert:**

Sweet alert è uno script Javasciprt che ho utilizzato per migliorare la visualizzazione degli alert.

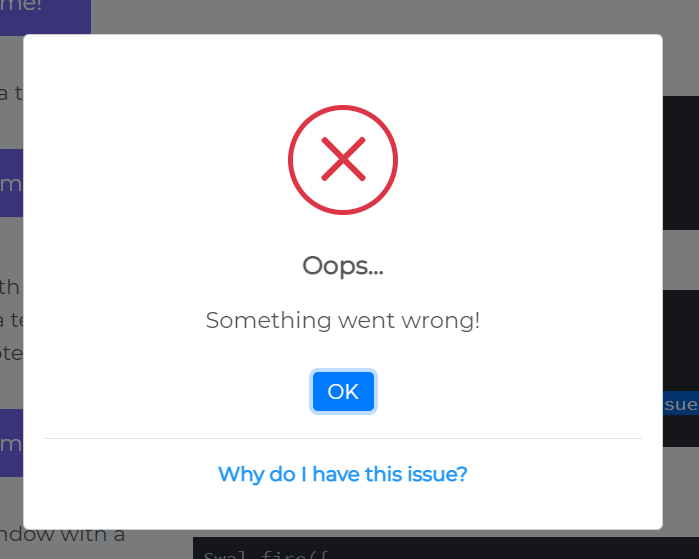
Per utilizzare questo script è necessaria la sua incorporazione nel codice della pagina tramite la seguente linea:

<script src="https://cdn.jsdelivr.net/npm/sweetalert2@10.16.6/dist/sweetalert2.all.min.js"></script>

Alert standard:



Sweet Alert:



Codice:

Swal.fire({

icon: 'error',

title: 'Oops...',

text: 'Something went wrong!',

footer: '<a href>Why do I have this issue?</a>'

})

**Linguaggio di markup utilizzato:**

Html “HyperText Markup Language” è un linguaggio di formattazione che descrive le modalità di impaginazione o visualizzazione grafica (layout) del contenuto, testuale e non, di una pagina web attraverso tag di formattazione.

Ogni tag (ad esempio <h1> o <p>) specifica un diverso ruolo dei contenuti che esso contrassegna (quindi il tag <h1> definirà un'importanza maggiore del tag <p>).

I browser che leggono il codice mostrano all'utente formattazioni predefinite per ogni tag che incontrano (così ad esempio i contenuti marcati con il tag <h1> avranno carattere 18pt e i contenuti marcati da <p> avranno carattere 12pt).

**Bootstrap:**

Bootstrap è una raccolta di strumenti liberi per la creazione di siti e applicazioni per il Web, contiene modelli di progettazione basati su HTML e CSS.

L’ho scelto rispetto ad un CMS in quanto offre una maggior libertà di scelta dell’implementazione, inoltre supporta anche il “Responsive web design”, ciò significa che il layout delle pagine web si regola dinamicamente, tenendo conto delle caratteristiche del dispositivo utilizzato, sia esso desktop, tablet o telefono cellulare.

Per utilizzare Bootstrap è necessario includere nel codice i collegamenti ad esse.

Es:

<link href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet">

**Css:**

Il CSS “Cascading Style Sheets” è un linguaggio usato per definire la formattazione di diversi documenti, nel mio caso l’ho utilizzato per la formattazione dei documenti Html e PHP.

Necessita di essere incorporato nel file in cui si richiede il suo utilizzo.

Es:

<link href="./style.css " rel="stylesheet" media="all">

**Phpseclib:**

Secure Communications Library offre una serie di file in formato PHP per l’implementazione di diverse soluzioni per gestire la criptazione e non solo.

Nel mio caso l’ho scelta per l’implementazione della criptazione RSA.

**Criptazione:**

Per prima cosa includo il file RSA.php fornitomi dalla libreria, mi salvo i valori passati tramite metodo post in delle variabili.

Successivamente inizializzo un oggetto **$rsa** legato alla classe **Crypt\_RSA** ed utilizzo la funzione **extract()** e il metodo **createKey()**.

**extract():**

Questa funzione utilizza chiavi dell'array come nomi di variabile ei valori come valori variabili.

Per ogni elemento si creerà una variabile nella tabella dei simboli.

**createKey():**

Ritorna un array con i seguenti elementi:

'privatekey': Chiave privata.

'publickey': Chiave pubblica.

Passo come parametro 2048 in quanto identifica la lunghezza in bit della chiave (2048 bit) in quanto sono lo standard di criptazione odierno.

****

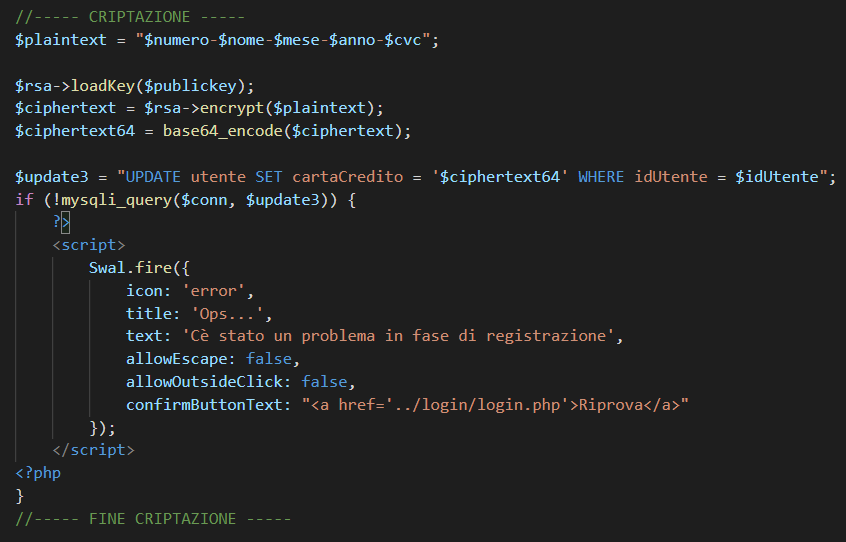
Per criptare il messaggio lo accodo le variabili separate da un carattere ‘-‘ in un’altra variabile chiamata **$plaintext** che conterrà il testo in chiaro.

Attraverso il metodo **loadKey()** a cui passo la chiave pubblica “carico” la chiave per la criptazione.

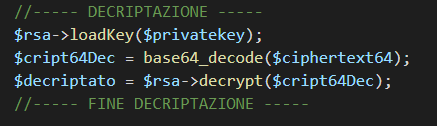
Cripto il testo in chiaro con il metodo **encrypt()** passandoglielo come parametro ed ottengo il testo criptato **$ciphertext**.

Una volta ottenuto il testo criptato ho la necessita di codificarlo in base 64 (**base64\_encode()**) in quanto il database non riesce a salvare il ciphertext nella sua versione originale binaria.

Successivamente eseguo la query per inserire i dati criptati e codificati (**$ciphertext64**) del database.



**Decriptazione:**

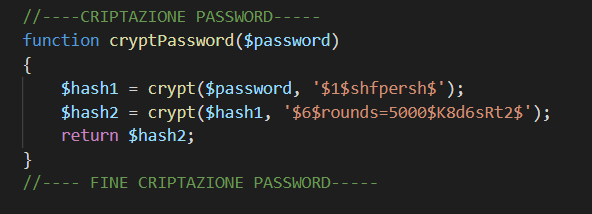
****

Per decriptare i dati utilizzo sempre il metodo **loadKey()** passandogli questa volta la chiave privata (**$privatekey**).

Riporto da base 64 a binario (**base64\_decode()**) e mi salvo il risultato nella variabile **$cript64Dec()**, infine decripto attraverso il metodo **decrypt()**.

Registrazione:

In fase di registrazione ho implementato la criptazione della password nel seguente modo:



La funzione riceve come parametro la password in chiaro inserita dall’utente, la passa alla funzione crypt().

Crypt necessita di due parametri la stringa da criptare e il metodo di criptazione con il salt ovvero dei caratteri aggiuntivi per migliorarne la sicurezza.

(La criptazione con SHA-512 ha bisogno di aver specificato l’ “hashing loop” indicato con rounds=500 ovvero il numero di volte che il ciclo hashing va ripetuto)

Nel primo caso $1$ identifica l’algoritmo MD5 con salt “shfpersh”.

Il risultato viene passato alla seconda funzione crypt la quale usa l’algorimo SHA-512 $6$ con salt.

**Controllo sulla password:**

In fase di inserimento della password ho verificato che venissero rispettate determinate caratteristiche:

Lunghezza minima di 8 caratteri

Minimo una lettera maiuscola

Minimo una lettera minuscola

Minimo un numero.

<input type="password" name="password" pattern="(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z]).{8,}" title="Almeno un numero, una lettara minuscola o Maiuscola e almeno 8 o più caratteri" placeholder="Password" required>