## Php

### Classe user:

#### Costruttore:

Per la gestione degli utenti lato php è stata creata una classe “*User*” che si occupa di verificare le varie informazioni legate all’utente. Il codice riportato sopra è la parte più importante della classe “*User*”. Come spiegato prima il costruttore da parametro riceve tutte le informazioni che gli servono per instanziare un nuovo utente. La classe è suddivisa in una parte a oggetti e una parta statica. Tutti i metodi che caretterizzano un utente sono a oggetti mentre quelli legati ai test sono statici, in seguito verrà mostrato un esempio.

        private $name;

        private $surname;

        private $birthday;

        private $city;

        private $address;

        private $houseNumber;

        private $telephoneNumber;

        private $email;

        private $gender;

        private $password;

        private $zipCode;

        /\*\*

        \* Costruttore personalizzato che si occupa di inizializzare gli attributi di un utente.

        \* Prima di inizializzare controlla che l'informazione sia corretta.

        \*/

public function \_\_construct($name,$surname,$birthday,$city,$zipCode,$address,$houseNumber,$telephoneNumber,$email,$gender,$password)

        {

            $this->name = User::tryName($name);

            $this->surname = User::tryName($surname);

            $this->birthday = User::tryDate($birthday);

            $this->city = User::tryName($city);

            $this->address = User::tryName($address);

            $this->houseNumber = User::tryHouseNumber($houseNumber);

            $this->telephoneNumber = User::tryNumber($telephoneNumber);

            $this->email = User::tryEmail($email);

            $this->gender = User::tryGender($gender);

            $this->zipCode = User::tryZipCode($zipCode);

            $this->password = User::tryPassword($password);

        }

#### Metodo di test:

Il seguente codice è un esempio per mostrare come funzionano i metodi di test. Come specificato prima il metodo è statico quindi può essere chiamato ovunque senza dover instaziare un nuovo *User.* Viene eseguito il codice di test, in questo caso viene verificato che sia più lungo di 8 caratteri e che ci sia almeno una lettera maiuscola. In seguito se il test è andato a buon fine allora si ritorna l’oggetto passato, altrimenti lancio una eccezione con un messagio personalizzato.

public static function tryPassword($object){

            if(strlen($object)>=8){

                if(strtolower($object) != $object){

                    return $object;

                }else{

                    throw new InvalidArgumentException(

sprintf( '"%s" have all lower character',$object)

);

                }

            }else{

                throw new InvalidArgumentException(sprintf( '"%s" is too short',$object));

            }

        }

### Classe Database:

#### Costruttore:

La classe per la connesione al database è composta da parecchi tool che possono essere utili. Prima di tutto nel costruttore della classe riceve tutte le informazioni necessarie per il giusto collegamento al database, come (nome del database, ip, porta, username, password). E istanzio una nuova connessione.

private $db;

        function \_\_construct($host,$port,$dbname,$username,$password)

        {

         $this->db = new PDO(

"mysql:host=$host;port=$port;dbname=$dbname", $username, $password

);

        }

#### Metodo per esecuzione query:

Questo metodo si occupa di eseguire e ritornare il risulato di una query. Il risultato che torna è già fetchato, oppure può anche non esserlo, quindi pronto all’uso. Nel caso la query che si sta andando a fare è sbagliata istanzio e lancio un’eccezione con messaggio personalizzato.

public function executeQuery($query){

            $result = $this->db->query($query);

            if ($result === FALSE) {

                throw new InvalidArgumentException(

"Failed to load schema is not exists or you are not permission");

            }

            return $result->fetchAll();

        }

        public function executeQueryWithoutFetch($query){

            $result = $this->db->query($query);

            if ($result === FALSE) {

                throw new InvalidArgumentException(

"Failed to load schema is not exists or you are not permission");

            }

            return $result;

        }

#### Metodo per stampare una tabella:

Una dei tool citati in precedenza è la possibilità di stampare una tabella già formattata per gli standard bootstrap. Il seguente metodo si occupa di stampare una tabella in base alla query che riceve.

public function printTableQuery($selectQuery){

            $result = $this->db->query($selectQuery);

            if ($result === FALSE) {

                throw new InvalidArgumentException(

"Failed to load schema is not exists or you are not permission");

            }

            $result = $result->fetchAll();

            echo "<table class='table' style='overflow-x:auto;'><thead><tr>";

            echo "<th></th>";

            echo "<th></th>";

                $n = 0;

                foreach ($result[0] as $key => $value) {

                    if($n%2==0){

                        echo "<th><b>".strtoupper($key{0}).substr($key,1,strlen($key))."</b></th>";

                    }

                    $n++;

                }

            echo "</tr></thead><tbody>";

            for ($i=0; $i < sizeof($result); $i++) {

                echo "<tr>";

 echo "<th><a href='php/database/modify.php?value=modify\_$i'><i class='fa fa-pencil' id='modify\_$i'></a></th>";

echo "<th><a href='php/database/modify.php?value=delete\_$i><i class='fa fa-trash' id='delete\_$i'></a></th>";

                for ($j=0; $j < sizeof($result[$i])/2; $j++) {

                    //echo "<tr><i class='far fa-trash-alt' id='$i'></tr>";

                    echo "<th>".$result[$i][$j]."</th>";

                }

                echo "</tr>";

            }

            echo "</tbody></table>";}

#### Metodo per aggiungere un utente:

Questo metodo tramite un parametro di tipo “*User*” si occupa di inserire nel database un utente e inviargli già la mail per la verifica.

public function insertUser($user){

            if(gettype($user) == "object"){

                if(get\_class($user) == "User"){

                    $name =$user->getName();

                    $birthday = $user->getBirthday();

                    $surname=$user->getSurname();

                    $street=$user->getAddress();

                    $house\_number=$user->getHouseNumber();

                    $zip\_code=$user->getZipCode();

                    $city=$user->getCity();

                    $email=$user->getEmail();

                    $phone\_number=$user->getTelephoneNumber();

                    $gender=$user->getGender();

                    $password=$user->getPassword();

                    $query = "Insert into user

                    (

                        name,

                        surname,

                        street,

                        house\_number,

                        zip\_code,

                        city,

                        email,

                        phone\_number,

                        gender,

                        password,

                        verified,

                        birthday,

                        type,

                        admin

                    )

                    values(

                        '$name',

                        '$surname',

                        '$street',

                        $house\_number,

                        $zip\_code,

                        '$city',

                        '$email',

                        $phone\_number,

                        '$gender',

                        '$password',

                        0,

                        $birthday,

                        'occasionale',

                        0

                    )";

                    $this->executeQuery($query);

                }else{

                    throw new InvalidArgumentException(get\_class($user)." is not a User class");

                }

            }else{

                throw new InvalidArgumentException(gettype($user)." is not a User class");

            }

        }

### Classe SendMail:

#### Costruttore:

Il costruttore di questa classe si occupa di stabilire un connesione con il server che fa da sender delle mail. Nel nostro caso abbiamo usato una libreria esterna per fare questo ovvero “*PHPMailer*” con relativa documentazione su github (<https://github.com/PHPMailer/PHPMailer/blob/master/README.md>).  
Inizialmente avevamo pensato di usare hotmail come server smtp, però dopo qualche mese di utilizzo abbiamo riscontrato troppi errori. Nella maggior parte delle volte hotmail bannava l’account e lo rendeva inutilizabile e bisognava sbloccarlo a mano ogni volta. Abbiamo quindi deciso di passare a un account gmail, che personalmente trovo più efficente.

public function \_\_construct()

        {

            $this->mail = new PHPMailer(true);

            $this->mail->isSMTP();

            $this->mail->Host = 'smtp.gmail.com';

            $this->mail->SMTPAuth = true;

            $this->mail->Username = 'gruppocasino2018@gmail.com';

            $this->mail->Password = 'Casin02018';

            $this->mail->SMTPSecure = 'tls';

            $this->mail->Port = 25;

            $this->mail->From = 'gruppocasino2018@gmail.com';

            $this->mail->FromName = 'Verify Password';

        }

#### Metodo mailSend:

Questo metodo si occupa semplicemente di spedire il messagio, in base ai parametri ricevuti.

        public function mailSend($email,$subject,$message){

            $this->mail->addAddress($email);

            $this->mail->isHTML(true);

            $this->mail->Subject = $subject;

            $this->mail->Body = $message;

            return $this->mail->send();

        }

### Esempio utilizzo classi:

#### Registrazione utente

$u = new User(

                    $\_POST["firstname"],

                    $\_POST["surname"],

                    $\_POST["birthday"],

                    $\_POST["city"],

                    $\_POST["zipCode"],

                    $\_POST["address"],

                    $\_POST["houseNumber"],

                    $\_POST["phoneNumber"],

                    $\_POST["email"],

                    $\_POST["gender"],

                    $\_POST["password"],

                    $\_POST["repassword"]

                );

                $db->insertUser($u);

                $cryptedMail = $\_POST["email"] ^ $privateKey;

                $message = '<hr>How are you? <br> <hr>This i your link:<a href="http://cashyland.tk/php/login/validate.php?id='.urlencode($cryptedMail).'">Click me!</a>';

                $subjet = "Hi there! Verify your email :)";

                $mailSender->mailSend($\_POST["email"],$subjet,$message);

                header("Location: ../../../verifyMail.html");

Test:

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-00x  REQ-001 | **Nome:** | Test classe User |
| **Descrizione:** | Questo va a verificare che la class user sia corretta. . | | |
| **Prerequisiti:** | Php e phpUnit installati | | |
| **Procedura:** | Verificare che la registrazione di utente sia corretta.   1. Provare a richiamare il metodo statico per la prova dell’email passandoli una mail valida 2. Provare a richiamare il metodo statico per la prova dell’email passandogli una mail non valida aspettandosi che quest’ultima tiri un exception 3. Provare a richiamare il metodo statico per la prova del nome passandoli un nome valido 4. Provare a richiamare il metodo statico per la prova del nome passandogli un nome non valida aspettandosi che quest’ultimo tiri un exception 5. Provare a richiamare il metodo che defisce l’età in base a una data passate. 6. Provare a richiamare il metodo statico per la prova della data passandoli una data valida 7. Provare a richiamare il metodo statico per la prova della data passandogli una data non valida aspettandosi che quest’ultima tiri un exception 8. Provare a richiamare il metodo statico per la prova del numero di casa passandogli un numero di casa valido 9. Provare a richiamare il metodo statico per la prova del numero di casa passandogli un numero di casa non valido aspettandosi che tiri un exception 10. Provare a richiamare il metodo statico per la prova del numero di telefono passandogli un telefono di casa valido 11. Provare a richiamare il metodo statico per la prova del numero di telefono passandogli un numero di telefono non valido aspettandosi che tiri un exception 12. Provare a richiamare il metodo statico per la prova del sesso passandogli un sesso valido 13. Provare a richiamare il metodo statico per la prova del sessopassandogli un sesso non valido aspettandosi che tiri un exception 14. Provare ad istanziare un nuovo user con dati corretti 15. Provare ad instanziare un nuovo con dati non corretti aspettandosi che quest’ultimo non venga istanziato. | | |
| **Risultati attesi:** | Non si presenta alcun problema legato a PHPUnit. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-00x  REQ-001 | **Nome:** | Test classe SendMail |
| **Descrizione:** | Questo va a verificare che la class SendMail sia corretta. . | | |
| **Prerequisiti:** | Php e phpUnit installati | | |
| **Procedura:** | Verificare che l’invio delle mail sia corretto.   1. Provare a inviare una mail mailSend(“[mail]”,”[subject]”,”[text]”) se la mail venisse spedita allora il metodo ritorna true. | | |
| **Risultati attesi:** | Non si presenta alcun problema legato a PHPUnit. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-00x  REQ-001 | **Nome:** | Test classe Database |
| **Descrizione:** | Questo va a verificare che la class Database sia corretta. . | | |
| **Prerequisiti:** | Php e phpUnit installati  La classe user funzionante TC-00x | | |
| **Procedura:** | Verificare che la connessione e l’eseguzione delle query sia corretta   1. Provare ad istanziare un oggetto database passandogli i dati legati al database 2. Provare ad eseguire una query con il metodo fornito dalla classe | | |
| **Risultati attesi:** | Non si presenta alcun problema legato a PHPUnit. | | |