Diario di lavoro

Luogo	Balerna
Data	14.05.2020

Lavori svolti

Ho riscontrato diversi problemi con la macchina che ho sto utilizzando per l'esame ma sono comunque riuscito a lavorare. Nella giornata di oggi mi sono occupato di implementare la gestione degli accessi interrogando il database MySQL. Il controllo dello stato di un utente verrà eseguito attraverso le sessioni di PHP. Per questo ho implementato una classe chiamata Session la quale mi permetterà di gestire le sessioni. Il metodo utilizzato per autenticare un utente è il seguente:

```php
public static function isAuthenticated() {
    return (isset($_SESSION["authenticated"]) && $_SESSION["authenticated"] == true);
}
```

Successivamente ho creato una classe per semplificare la gestione dei permessi chiamata Permission nella quale ho salvato in delle costanti i valori dei permessi:

```php
const INSERT = 1;
const SELECT = 2;
const CREATE = 4;
const ADMINISTRATOR = 8;
```

E successivamente ho implementato un metodo per ogni permesso, ad esempio canInsert:

```php
public static function canInsert($permission = null) {
    if ($permission === null) {
        $permission = $_SESSION["permission"];
    }
    return ($permission & self::INSERT) === self::INSERT || self::isAdmin($permission);
}
```

In questo modo potrò verificare i permessi di ogni utente sia attraverso la sessione che tramite parametro del metodo. Ho implementato anche dei validatori di base nella classe Validator, come ad esempio per verificare che un nome sia valido:

```php
public static function isValidName($name) {
    return preg_match('/^[A-Za-zÀ-ÖØ-öø-ÿ]{1,20}$/', $name);
}
```

Ho implementato validazione per nome, cognome, password, email e numeri. Le classi implementate in precedenza le ho utilizzate all'interno dei middleware AdminRequired e AuthRequired i quali verranno utilizzati per dare delle condizioni di accesso ai vari percorsi dell'applicativo. Ad esempio il codice di AuthRequired utilizza la classe Session per controllare se l'utente ha eseguito l'accesso:

```php
class AuthRequired {
    public function __invoke($request, $response) {
        if (!Session::isAuthenticated()) {
            $response->redirect("/login");
            exit;
        }
    }
}
```

In questo modo mi basterà assegnare queste classi ai percorsi per determinare i permessi di accesso. Successivamente ho creato una classe Model la quale verrà utilizzata per verificare la presenza degli utenti all'interno della banca dati, ho quindi implementato diversi metodi tra cui:

```php
public static function getByEmail($email) {
    $pdo = Database::getConnection();
    $query = "SELECT * FROM user WHERE email = :email";
    $stm = $pdo->prepare($query);
    $stm->bindParam(':email', $email);
    try {
        $stm->execute();
        return $stm->fetch(\PDO::FETCH_ASSOC);
    } catch (\PDOException $e) {
        return false;
    }
}
```

Questo metodo verrà utilizzato dal login per verificare la presenza di un utente. Ho dunque creato anche un controller chiamato Auth il qual andrà a gestire tutti i percorsi e le richieste relative agli accessi nel quale ho implementato metodi per caricare delle viste di prova, come ad esempio quella di login:

```php
public static function login(Request $req, Response $res) {
    return $res->render(__DIR__ . '/../Views/Auth/login.php');
}
```

Successivamente ho implementato anche il metodo per eseguire il login:

```php
public static function doLogin(Request $req, Response $res) {
    $email = $req->getParam("email");
    $password = $req->getParam("password");
    if (isset($email) && Validator::isValidEmail($email) && isset($password)) {
        $user = Users::getByEmail($email);
        if ($user && password_verify($password, $user["password"])) {
            unset($user["password"]);
            Session::authenticate($user);
            return $res->withStatus(200);
        }
    }
    return $res->withStatus(403);
}
```

Il login verrà verificato attraverso gli status code della risposta. Come ultima cosa ho aggiunto dei percorsi al router del progetto:

```php
// Percorso pagina di accesso.
$router->get("/login", "FilippoFinke\Controllers\Auth::login");
// Percorso pagina di recupero password.
$router->get("/forgot-password", "FilippoFinke\Controllers\Auth::forgotPassword");
// Percorso per eseguire il login.
$router->post("/login", "FilippoFinke\Controllers\Auth::doLogin");
// Percorso per eseguire la disconnessione.
$router->get("/logout", "FilippoFinke\Controllers\Auth::doLogout");
// Gruppo di percorsi dove è richiesto solamente l'accesso.
$homeRoutes = new RouteGroup();
$homeRoutes->add(
    // Percorso pagina principale.
    $router->get("/", function($req, $res) {
        return $res->withHtml("<a href='/logout'>Esci</a>");
    })
)
// Controllo autenticazione.
->before(new AuthRequired());
```

In questo modo ho potuto eseguire dei test sul funzionamento del login.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato nella giornata di oggi.

Punto della situazione rispetto alla pianificazione

Al momento mi trovo in linea con la pianificazione essendo che ho terminato la gestione dell'accesso all'applicativo web.

Programma di massima per la prossima giornata di lavoro

Sviluppare la funzionalità di recupero password ed iniziare a sviluppare la gestione utenti.