

# Diario di lavoro

Luogo	Scuola Arti e Mestieri Trevano
Data	17.02.2020

## Lavori svolti

Durante la giornata di oggi ho concluso tutta la parte che concerne il login. Infatti oggi ho concluso la parte di recupero della password e ho iniziato e concluso il form di login. La prima parte di codice che ho scritto è l'identificazione dell'utente quando clicca il link di recupero password. Questa fase è molto simile alla parte di registrazione, in quanto utilizzo un hash e la email dell'utente per identificare univocamente la persona che vuole cambiare la password. Il metodo che esegue questa azione si chiama `resetPassword()` e appartiene al controller "CambiaPassword":

```
public function resetPassword($hash = null,$email= null){
    if(($email != null) && ($hash != null)) {
        require_once 'application/models/confermaModel.php';
        $conferma_model = new ConfermaModel();
        if($conferma_model->changePassword(Util::test_input($hash),
Util::test_input($email)){
            $_SESSION[SESSION_CHANGE_PASSWORD] = Util::test_input($email);
            header("Location: " . URL . "cambiaPassword");
        }else{
            header("Location: " . URL . "errore");
        }
    }else{
        header("Location: " . URL . "errore");
    }
}
```

che a sua volta richiama il metodo `changePassword()` del model "CambiaPasswordModel":

```
function changePassword($hash,$email){
    $selectUsers = "SELECT * FROM utente WHERE email=%s AND hash_email=%s";
    $result = $this->connection->query($selectUsers, $email, $hash);
    if($result != null) {
        $updateUsers = "UPDATE utente SET hash_email='0' WHERE email=%s";
        $this->connection->query($updateUsers, $email);
        return true;
    }else{
        return false;
    }
}
```

Una volta identificato l'utente, quest'ultimo inserisce la nuova password due volte e clicca il bottone "Cambia" (Vedi scorso diario).

Il controller che si occupa di modificare si chiama modifyPassword() e appartiene sempre al controller "CambiaPassword":

```
public function modifyPassword(){
    require 'application/models/cambiaPasswordModel.php';
    $cpm = new CambiaPasswordModel();
    if($_SERVER["REQUEST_METHOD"] == "POST"){
        if($cpm->
>modify($_SESSION[SESSION_CHANGE_PASSWORD],Util::test_input($_POST[PASSWORD]),
Util::test_input($_POST[RE_PASSWORD]))){
            require 'application/libs/emptySession.php';
            EmptySession::changePass();
            header("Location: ".URL."passwordCambiata");
        }else{
            header("Location: ".URL."errore");
        }
    }else{
        header("Location: " . URL . "errore");
    }
}
```

che a sua volta richiama il metodo modify() del model "CambiaPasswordModel"

```
public function modify($email, $password1, $password2){
    echo $email." ".$password1." ".$password2;
    if($this->validator->checkPassword($password1,$password2)){
        $password = password_hash($password1,PASSWORD_DEFAULT);
        $updateUsers = "UPDATE utente SET password=%s WHERE email=%s";
        $this->connection->query($updateUsers, $password, $email);
        return true;
    }
    return false;
}
```

Una volta finite la modifica della password ho creato il back-end della pagina di login in ajax. Tutto parte da questo pezzo di Javascript:

```
function login() {
    var inputs = document.getElementsByTagName("input");
    var v = new Validator();
    $.when(
        v.accessLogin(inputs[0].value, inputs[1].value)
    ).done( function (json) {
        var response = json;
        if(response == 1){
            document.getElementById("alert_validation_login").style.display = "none";
            document.getElementById("alert_validation_login_s").style.display = "block";
            window.setTimeout(function() {
                window.location.replace(URL);
            }, 1000);
        }else if(response == 2){
            window.location.replace(URL + "verificaEmail");
        }else{
            document.getElementById("alert_validation_login").style.display = "block";
            document.getElementById("alert_validation_login_s").style.display = "none";
        }
    });
}
```

A sua volta richiama il metodo accessLogin della mia classe Validator di JavaScript:

```
accessLogin(email_val,password_val) {
    var email = email_val;
    var password = password_val;
    var response;
    return ($.ajax({
        url: (URL + "login/access"),
        type: "POST",
        dataType: "json",
        data: {email: email, password: password},
        success: function (text) {
            response = text;
        }
    }));
}
```

In questo metodo preparo la richiesta in ajax con il metodo post dove invio, al metodo access() del controller login, l'email e la password.

Il metodo access() si presenta così:

```
public function access() {
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $email = Util::test_input($_POST[EMAIL]);
        $password = $_POST[PASSWORD];
        require_once 'application/models/loginModel.php';
        $loginModel = new LoginModel();
        $result = $loginModel->access($email,$password);
        if ($result != null){
            $_SESSION[SESSION_TYPE] = $result[0][DB_USER_TYPE];
            $_SESSION[SESSION_EMAIL] = $email;
        }
    }else{
        header("Location:" . URL . "errore");
    }
}
```

Che a sua volta richiama il metodo access() del model LoginMode:

```
public function access($email, $password) {
    $result = $this->getUser($email);
    if (count($result) > 0) {
        if (password_verify($password, $result[0][DB_USER_PASSWORD])) {
            if ($result[0][DB_USER_TYPE] & 1) {
                echo 1;
                return $result; //Credenziali corrette e verificate
            } else {
                echo 2; //Credenziali corrette ma non verificate
            }
        } else {
            echo 0; // Credenziali sbagliate
        }
    } else {
        echo 0;
    }
}
```

Se l'utente che ha eseguito l'accesso non ha ancora verificato la email verrà portato in questa pagina:



## Verifica la tua email

Non puoi accedere al sito se non verifichi la tua email. Perciò ti preghiamo di cliccare link nella email che ti abbiamo inviato dopo la registrazione.

[Torna al login](#)

Se le credenziali dell'utente sono errate si presenta così:



## Login

Email o password errati!

asd



....

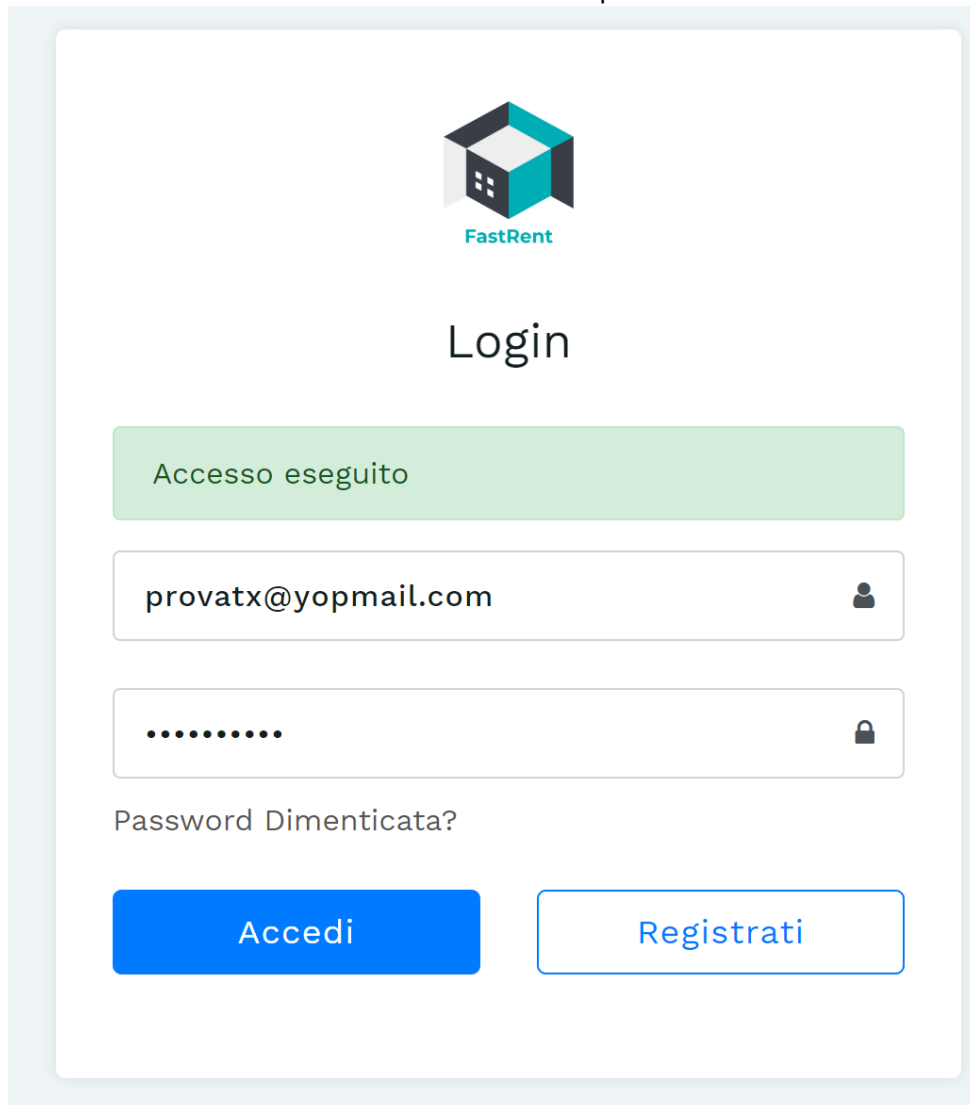


[Password Dimenticata?](#)

[Accedi](#)

[Registrati](#)

Infine se le credenziali di accesso sono corrette si presenta così:



The image shows a login page for 'FastRent'. At the top is the FastRent logo, which consists of a stylized 3D cube icon with the text 'FastRent' below it. Below the logo is the word 'Login' in a large, black, sans-serif font. Underneath 'Login' is a light green rectangular box containing the text 'Accesso eseguito'. Below this box are two input fields. The first input field contains the email address 'provatx@yopmail.com' and has a user icon on the right. The second input field contains a series of dots representing a password and has a lock icon on the right. Below the password field is the text 'Password Dimenticata?'. At the bottom of the form are two buttons: a solid blue button labeled 'Accedi' and a white button with a blue border labeled 'Registrati'.

e dopo 1 secondo l'utente viene mandato alla pagina principale. La pagina principale l'ho incominciata a creare ma la finirò la prossima lezione.

#### Problemi riscontrati e soluzioni adottate

Durante la giornata di oggi non ho avuto particolari problemi.

#### Punto della situazione rispetto alla pianificazione

Sono in anticipo di 4 ore rispetto alla mia pianificazione

#### Programma di massima per la prossima giornata di lavoro

Durante la prossima giornata completerò il front-end della pagina principale. Se rimane tempo inizierò la pagina di gestione utenti.