Validation trong Laravel

Giảng viên: Bùi Quang Đăng

Contents

Stanford

Laravel Framework

LÀM VIỆC VỚI VALIDATION LARAVEL

Stanford
nâng tầm tri thức

Giới thiệu về Validation trong Laravel

- Validation là một công cụ do Laravel cung cấp dùng để validate (kiểm tra) dữ liệu request được gửi lên trước khi nó được xử lý.
- Validate dữ liệu giúp chúng ta đảm bảo được tính đúng đắn của dữ liệu, bắt những lỗi nhập sai dữ liệu của người dùng, bảo vệ hệ thống khỏi nhiều dạng tấn công điển hình như SQL Injection. Khi đảm bảo được dữ liệu là đúng đắn, hệ thống sẽ hoạt động ổn định, ít xảy ra lỗi.

Stanford
nâng tầm tri thức

Làm việc với Validation trong Laravel

- Khi các parameters không vượt qua được các validation rules, Laravel sẽ tự động chuyển hướng người dùng quay lại. Tất cả các validation erors message sẽ tự động trả về kèm theo dưới dạng flash session.

❖ Một số Validation hay sử dụng trong Laravel

Tên validation	Mô tả
required	Yêu cầu không được bỏ trống thông tin
bail	Dừng kiểm tra nếu một điều kiện kiểm tra bị vi phạm
between:min,max	Dữ liệu kiểm tra cần trong khoảng min, max
confirmed	Sử dụng để so khớp với thông tin khác theo cấu trúc {field}_confirmation, ví dụ thông tin password thì trường password_confirmation cần so khớp với password trên
date	Kiểm tra kiểu ngày tháng hợp lệ
date_equals:date	So sánh bằng ngày cần so khớp
date_format:format	Kiểm tra đúng định dạng cần so khớp
alpha	Yêu cầu nội dung là dạng chỉ chứa kí tự

❖ Một số Validation hay sử dụng trong Laravel

Tên validation	Mô tả
alpha_num	Yêu cầu nội dung là kí tự chữ
array	Yêu cầu dữ liệu là dạng mảng
email	Kiểm tra định dạng email
number	Kiểm tra thông tin nhập vào phải là kiểu số
integer	Kiểm tra thông tin là kiểu số nguyên
nullable	Trường thông tin kiểm tra có thể null
size:value	Kiểm tra dữ liệu cần có độ dài bao nhiêu
same:field	Kiểm tra trùng với đối tượng cần so khớp
string	Kiểm tra nội dung là dạng chuỗi
Regex:pattern	Nội dung thoả mãn theo mẫu pattern định nghĩa, ví dụ 'email' => 'regex:/^.+@.+$/i'

Làm việc với Validation trong Laravel

- Định nghĩa các thông tin kiểm tra bắt lỗi và thông báo

```
//Kiểm tra nhập thông tin
$rules = [
    'tenDangNhap'=>'required',
    'matKhau'=>'required'
];
$message = ['tenDangNhap.required'=>'Bạn cần nhập tên đăng
nhập',
'matKhau.required'=>'Bạn cần phải nhập mật khẩu đăng nhập'];

$validator = $request->validate($rules,$message);
```

Làm việc với Validation trong Laravel

- Xử lý hiển thị thông tin lỗi trên giao diện

```
@if($errors->any())
<div class="alert alert-danger" role="alert">
    @foreach($errors->all() as $error)
        {{ $error }}<br/>
    @endforeach
</div>
@endif
```

Làm việc với Validation trong Laravel

- Sử dụng hàm Validator::**make**()

```
Validator::make($input, $rules,
$messages, $attributes);
```

- Trong đó:
  - **$input** là các đối tượng cần validation
  - **$rules** chứa các thông tin cần validation
  - **$messages** nội dung các thông báo khi vi phạm lỗi validation
  - **$attributes** chứa các thông tin mô tả tên của các trường tương ứng

Làm việc với Validation trong Laravel

- Sử dụng hàm Validator::**make**()

```
$input = $request->all();
$rules = [
    'TenSach' => 'required|max:250',
    'GiaSach' => 'required|numeric',
    'TacGia'=>'required|max:30'
];
$messages = [
'TenSach.required' => 'Bạn cần phải nhập tên sách',
'GiaSach.required' => 'Bạn cần phải nhập giá sách',
'GiaSach.numeric'=>'Bạn cần phải nhập giá sách kiểu số',
'TacGia.required'=>'Bạn cần phải nhập tên tác giả',
'TacGia.max'=>'Bạn cần được nhập tác giả quá 30 kí tự'
];
```

Làm việc với Validation trong Laravel

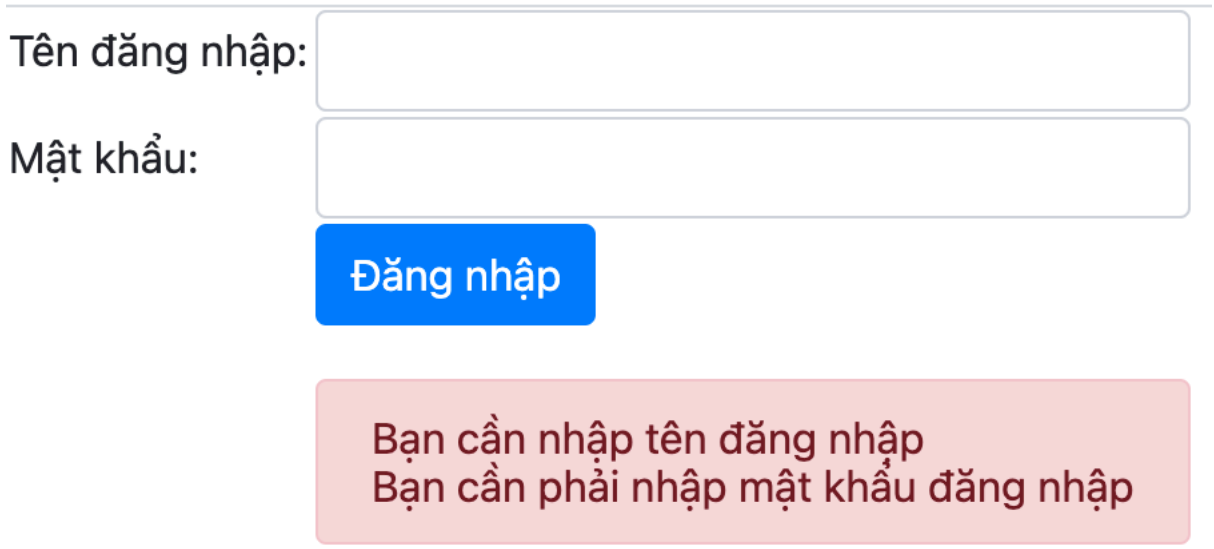
- Sử dụng hàm Validator::**make**()

```
$attributes = ['tenDangNhap'=>'tên đăng nhập',
               'matKhau'=>'mật khẩu'];
$validator = Validator::make($input, $rules, $messages,
$attributes);
$request->session()->flash('message', 'Bạn cần phải nhập đầy đủ
thông tin');
if ($validator->fails()) {

    return view('sach.themmoi')
                ->withErrors($validator);
}
else{
```

Làm việc với Validation trong Laravel

- Giao diện khi không nhập thông tin



LÀM VIỆC VỚI FORM REQUEST

Làm việc với Form Requests trong Laravel

- Form request thực chất là việc bạn tách riêng phần validation và phần code xử lý. Bạn sẽ tạo 1 file riêng phục vụ cho mục đích validate dữ liệu còn file controller sẽ chỉ chứa code xử lý. Làm như vậy sẽ giúp code của bạn phân chia rõ ràng, sạch sẽ và dễ đọc hơn.
- Cú pháp để tạo form request như sau:

```
php artisan make:request [Tên form request]
```

Làm việc với Form Requests trong Laravel

- Xử lý validation trong lớp Request tạo mới

```php
namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class DangNhapRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }
```

Làm việc với Form Requests trong Laravel

- Xử lý validation trong lớp Request tạo mới

```
/**
 * Khai báo các validation trong hàm
 *
 * @return array
 */
public function rules()
{
    return [
        'tenDangNhap'=>'required',
        'matKhau'=>'required'
    ];
}
```

Làm việc với Form Requests trong Laravel

- Định nghĩa các thông báo trong lớp Request tạo mới

```
/**
 * Khai báo các thông báo lỗi tương ứng
 */
public function messages()
{
    return ['tenDangNhap.required'=>'Bạn cần nhập tên đăng nhập',
    'matKhau.required'=>'Bạn cần phải nhập mật khẩu đăng nhập'];
}
```

Làm việc với Form Requests trong Laravel

- Định nghĩa các thông báo trong lớp Request tạo mới

```
/**
 * Định nghĩa tên cho các thuộc tính
 */
public function attributes()
{
    return ['tenDangNhap'=> 'tên đăng nhập',
            'matKhau'=> 'mật khẩu'];
}
```

Làm việc với Form Requests trong Laravel

- Sử dụng trong lớp Controller tương ứng để bắt lỗi

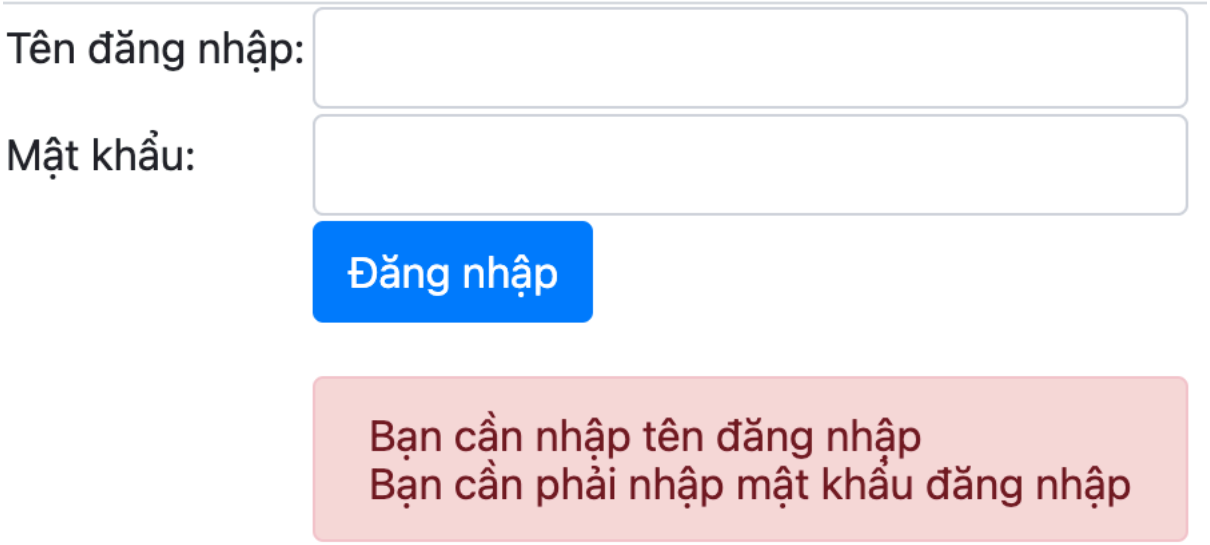
```
use App\Http\Requests\DangNhapRequest;
```

```
public function xuLyDangNhap2(DangNhapRequest $request)
{
    $tenDangNhap = $request->input('tenDangNhap');
    $matKhau = $request->input('matKhau');

    echo "Tên đăng nhập: " . $tenDangNhap . "<br>";
    echo "Mật khẩu: " . $matKhau . "<br>";
    return response()->view("DangNhap");
}
```

Làm việc với Form Requests trong Laravel

- Giao diện khi người dùng không nhập thông tin



Làm việc với Form Requests trong Laravel

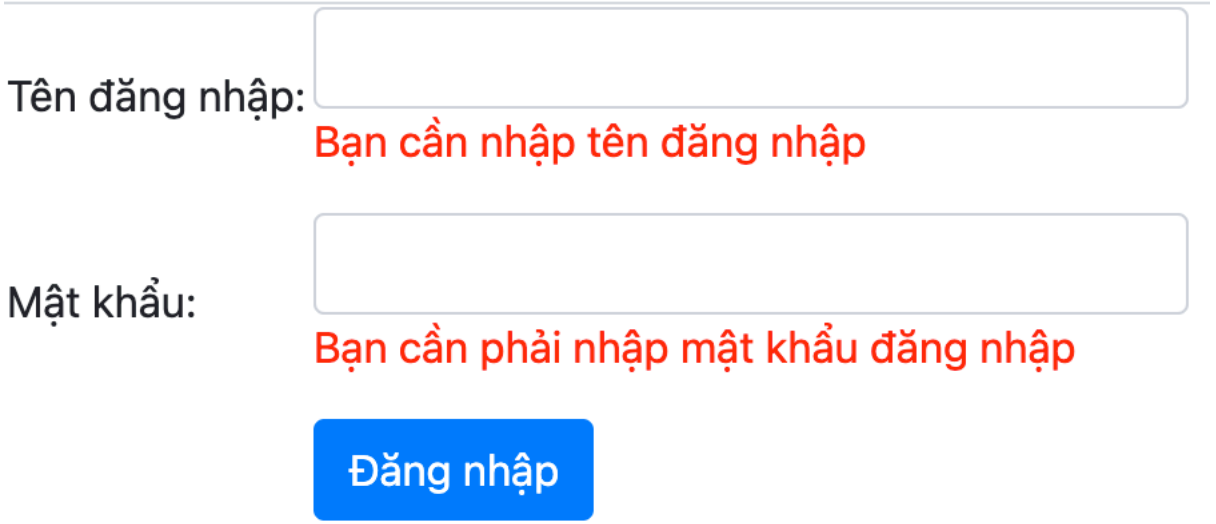
- Xử lý lỗi theo từng trường thông tin

```
<input type="text" name="tenDangNhap" class="form-control"/>
 @if ($errors->has('tenDangNhap'))
<p class="errorinfo">{{ $errors->first('tenDangNhap') }}</p>
@endif
```

```
 <input type="password" name="matKhau" class="form-control"/>
 @if ($errors->has('matKhau'))
<p class="errorinfo">{{ $errors->first('matKhau') }}</p>
@endif
```

Làm việc với Form Requests trong Laravel

- Xử lý lỗi theo từng trường thông tin



LÀM VIỆC VỚI CLOSURES, RULES

Sử dụng Closures trong Laravel

- Đầu tiên cần khai báo các điều kiện của các fields là dạng array

```
public function rules() {
return [ 'name' => [ 'required', 'size:6', ],
'number' => [ 'required', 'numeric' ] ]; }
```

- Thêm closure vào các field theo cấu trúc sau:

```
function ($attribute, $value, $fail)
{
}
```

Sử dụng Closures trong Laravel

- **Closure** trên gồm có 3 biến mặc định là:
  - **$attribute**: chính là tên của field cần validate tương ứng, ở đây sẽ lần lượt là name và number.
  - **$value**: là giá trị nhận vào khi người dùng submit form.
  - **$fail**: là một callback được gọi đến khi việc validate thất bại. Sử dụng để truyền thông báo khi thông tin không hợp lệ.

Stanford

Sử dụng Closures trong Laravel

```
public function rules() {
return [
'name' => [ 'required', 'size:6’,
function ($attribute, $value, $fail)
{
if (strtoupper($value) !== $value)
{
return $fail(”Trường $attribute cần nhập kí tự viết hoa");
}
} ],
'number' => [ 'required', 'numeric’
function ($attribute, $value, $fail)
{
} ]
];
}
```

Rule Object trong Laravel

- Rule Object là một class mà trong đó chúng ta có thể định nghĩa các custom rule của bạn tương tự như Closure và có thể gọi đến instance của lớp rule đó ở bất cứ đâu khi cần sử dụng.
- **Cú pháp tạo rule mới:**

```
php artisan make:rule [Tên rule mới]
```

Stanford

Rule Object trong Laravel

- Class tạo ra bao gồm có 3 hàm:
  - **__construct()**: hàm khởi tạo của rule, nơi ta có thể truyền thêm các biến khác vào
  - **passes()**: hàm để định nghĩa giá trị của filed cần vali có thỏa mãn hay không (2 biến $attribute và $value có giá trị tương tự như sử dụng Closure)
  - **mesage()**: thông báo trả về nếu điều kiện không thỏa mãn

Stanford
nâng tầm tri thức

Rule Object trong Laravel

- Xử lý trên lớp Rule tạo ra

```php
namespace App\Rules;

use Illuminate\Contracts\Validation\Rule;

class CustomRequireUpper implements Rule
{
    /**
     * Create a new rule instance.
     *
     * @return void
     */
    public function __construct()
    {
    }
```

Rule Object trong Laravel

- Xử lý trên lớp Rule tạo ra

```php
/**
 * Determine if the validation rule passes.
 *
 * @param  string  $attribute Tên thuộc tính
 * @param  mixed  $value Giá trị nhận được khi submit
 * @return bool
 */
public function passes($attribute, $value)
{
    return strtoupper($value) == $value;
}
```

Rule Object trong Laravel

- Xử lý trên lớp Rule tạo ra

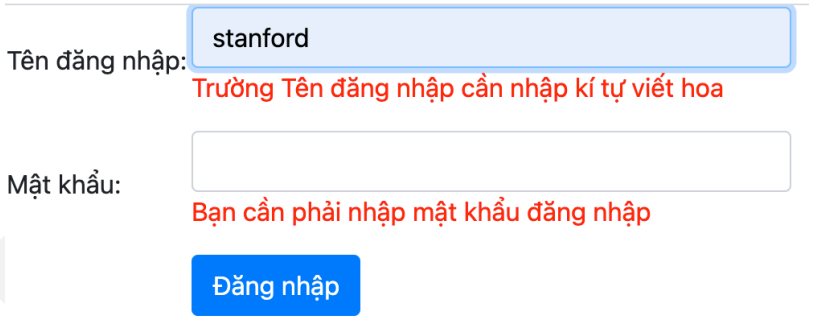
```
/**
 * Nội dung thông báo khi bị lỗi
 *
 * @return string
 */
public function message()
{
    return 'Trường :attribute cần nhập kí tự viết hoa';
}
```

Rule Object trong Laravel

- Sử dụng Rule vừa định nghĩa

```php
use App\Rules\CustomRequireUpper;
```

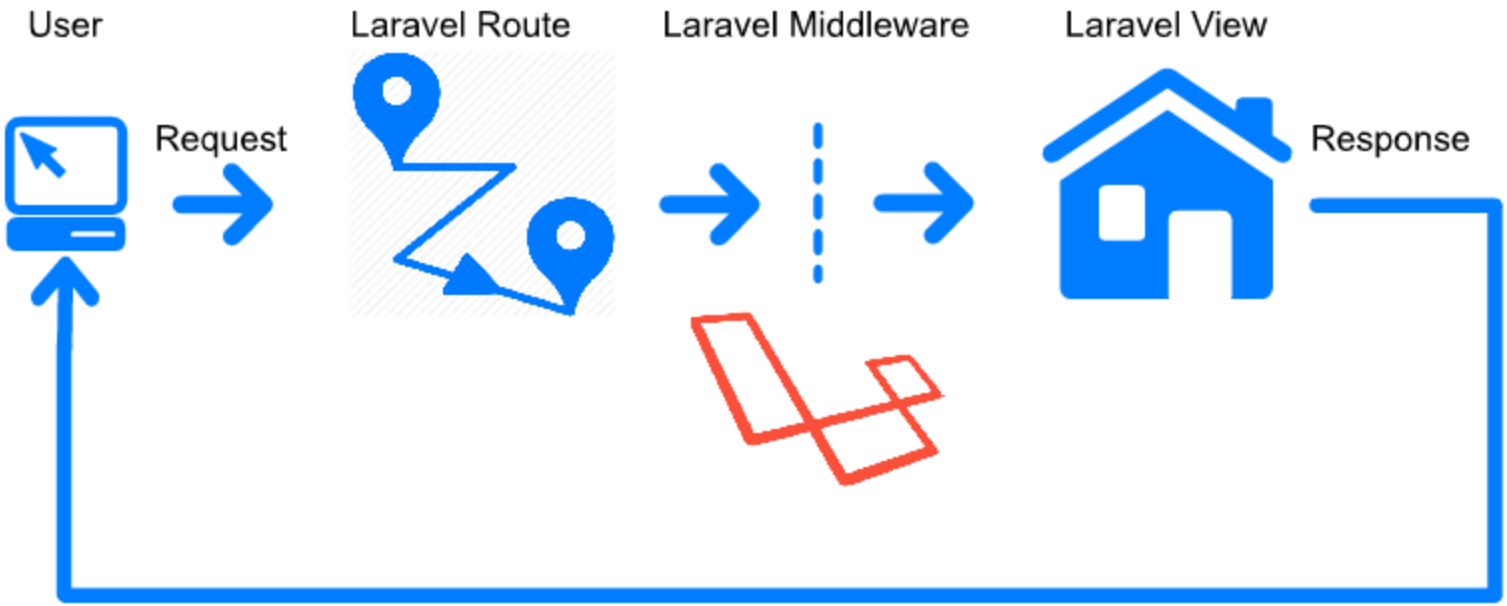
```php
/**
 * Khai báo các validation trong hàm
 *
 * @return array
 */
public function rules()
{
    return [
        'tenDangNhap'=>['required',
        new CustomRequireUpper()
    ],
        'matKhau'=>'required'
    ];
}
```



LÀM VIỆC VỚI MIDDLEWARE

Làm việc với Laravel Middleware

- Middleware đúng như tên gọi của nó, là đoạn code trung gian đứng giữa request và response, cung cấp một cơ chế bộ lọc.



Làm việc với Laravel Middleware

- Middleware đúng như tên gọi của nó, là đoạn code trung gian đứng giữa request và response, cung cấp một cơ chế bộ lọc.
  - Ví dụ: Laravel cung cấp một middleware để xác nhận xem một người dùng đã xác thực chưa, nếu người dùng đã qua xác thực sẽ được chuyển hướng đến trang quản trị hoặc nếu chưa xác thực sẽ được chuyển đến trang đăng nhập

Stanford
nâng tầm tri thức

Làm việc với Laravel Middleware

- Tất cả các middleware đều nằm trong thư mục app/Http/Middleware.
- Cú pháp để tạo một middleware mới như sau:

```
php artisan make:middleware <Tên middleware>
```

Stanford

Làm việc với Laravel Middleware

- Xử lý thông tin trong hàm handle:

```
public function handle(Request $request, Closure $next)
{
    if($request->session()->get('userOnline')== null)
    {
        return redirect('/dang-nhap');
    }
    return $next($request);
}
```

Làm việc với Laravel Middleware

- Xử lý thông trong hàm handle:

```
public function handle(Request $request, Closure $next, $role)
{
    echo "Vai trò là: " . $role;

    if($role != 'admin')
    {
        return redirect('/login1');
    }
    return $next($request);
}
```

Làm việc với Laravel Middleware

- Thực hiện đăng ký sử dụng Middleware:
  - **Global middleware** là middleware sẽ được sử dụng với bất kể yêu cầu HTTP đến hệ thống, đơn giản là thêm middleware này vào thuộc tính $middleware trong class **app/Http/Kernel.php**.
  - **Route middleware** là gắn một middleware với một route xác định, trước khi gắn một route với một middleware phải liệt kê chúng vào thuộc tính $routeMiddleware trong class app/Http/Kernel.php.

Stanford

Làm việc với Laravel Middleware

- Thực hiện đăng ký sử dụng Middleware:

```
protected $routeMiddleware = [
    'auth' => \App\Http\Middleware\Authenticate::class,
    'auth.basic' =>
    \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
    \Illuminate\Routing\Middleware\ThrottleRequests::class,
    'verified' =>
    \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
    'kiemTraDangNhap'=>\App\Http\Middleware\XuLyDangNhap::class,
];
```

- Sử dụng Middleware với router cần:

```
Route::get('/qlsach', [SachController::class,'layDanhSach'])
->middleware('kiemTraDangNhap:admin1');
```

Làm việc với Laravel Middleware

- Khi vào trang quản lý sách sẽ yêu cầu đăng nhập xác thực thành công thì mới vào trang danh sách:

Danh sách thông tin sách

Thêm mới

Ảnh	Id	Tên sách	Mô tả	Giá	Ngày tạo	Tác giả	
	1	Đắc Nhân Tâm	Sách dạy về kỹ năng sống để thành đạt hơn trong cuộc sống của mỗi con người	250000	2020-12-02	Dale Carnegie	Sửa Xoá
	2	Cha giàu cha nghèo	Sách nói về kỹ năng quản lý tài chính của 2 người cha giàu và cha nghèo của tác giả	200000	2020-12-02	Robert T.Kiyosaki	Sửa Xoá
	3	Lập trình C++	Sách dạy về ngôn ngữ lập trình c++ cho người mới học	150000	2020-12-02	Phạm Văn Ất	Sửa Xoá
	4	Trí tuệ nhân tạo	Sách nói về trí tuệ nhân tạo áp dụng trong cuộc sống hiện tại	200000	2020-12-07	Sim mon	Sửa Xoá

Tên đăng nhập:

Mật khẩu:

Đăng nhập

Exercises


Stanford
nâng tầm tri thức
Thank You !
www.stanford.com.vn