API CAT 2

102730

10/4/2020

Strathmore University

BBT 3102 –Application Programming for the Internet

CAT II – Practical CAT – 30 Marks

% Weight: 8%

Have composer installed in your machine.

You must have a fresh copy of Laravel (V5.7 and above) installed in a folder (Folder name should be your student number) your machine:

Ensure MySQL Database is installed in your machine (if you have Xampp installed in your machine, then you're fine.)

Add a new user in your MySQL database (username: your student number, password: a password of your choice). Note that you will not use default user 'root' in this work! This time your database user will be the new user you just created!

The server to be used in this work is assumed to:

Database server: MySQL; port number: Default-3306

Web server: Localhost (your machine) will host Laravel webserver running on default port 8000

Project narrative:

Think of the relationships between a 'student' and 'fees payment' in a school environment. There is a form used to register a student (with details, student number, full name, date of birth and address) and another form used to record a fee payment (with details student that is paying fees, date of payment and amount) instance. Because it is not possible for all the students to pay their fees once, they are allowed to pay in 'bits'. This means one student pays fees once or many times.

Required:

1. Set your database variables in your .env file
2. In your Laravel folder structure, create a folder (the name of this folder should be your student name) in views folder. This folder will hold all your views in this work.
3. Create 2 views to hold forms for registering a new student and for paying fees. Name the views as *student.blade.php* and *fees.blade.php* respectively.
4. Create a home view (give name of your choice) such that when one visits 127.0.0.1:8000, they see a view with 2 links as follows:
   - Student – When clicked, shows a form used to register a new student
   - Fees – when clicked, shows a form used to record new fees payment by a student.
5. Create two models (from Laravel command line) Student and Fees together with their migration files. Complete the migration files to show the all the data they will store. Use appropriate data types.
   - Record the Laravel commands you used to create your models
   - Show how the migration files looks like after you update
   - Which commands do you use to finally create your table?
6. Write two methods, one in each model to bring out this relationship, 'Student owns many fees payments and that fees payments belongs to Students'. Show the methods.
7. Create a controller for each model. Which command did you use?
8. When data in the form are submitted, it is saved as follows
   - Students form saves data into a table called 'students'
   - Fees payment forms saves data into fees table.

- All form data must be validated in Laravel

9. Disclose other information by filling the tables below. An example is given in all routes

All Routes:

Complete Route
Route::get('/games/all', 'GamesController@allGames');
`Route::middleware('auth:api')->get('/user', function (Request $request) {` `    return $request->user();`

All database tables:

Names of table

`Schema::create('users', function (Blueprint $table) {`
`$table->id();`
`$table->string('name');`
`$table->string('email')->unique();`
`$table->timestamp('email_verified_at')->nullable();`
`$table->string('password');`
`$table->rememberToken();`
`$table->timestamps();`
`});`
`}`
`public function down()`
`{`
`Schema::dropIfExists('users');`
`}`
`}`

`{`
`Schema::create('password_resets', function (Blueprint $table) {`
`$table->string('email')->index();`
`$table->string('token');`
`$table->timestamp('created_at')->nullable();`
`});`
`}`

All controllers:

Name of model

```
class FeesController extends Controller
{
    public function index()
    {
        $students = Student::all();
        return view('102730.fees')->with('students',$students);

    }

    public function storePayment(Request $request)
    {
        $this->validate($request, [
            'student_number'=>'required',
            'date_of_payment'=>'required',
            'amount'=>'required'
        ]);

        $fee = new Fee();
        $fee->student_number = $request->student_number;
        $fee->date_of_payment = $request->date_of_payment;
        $fee->amount = $request->amount;
        $fee->save();

        return redirect()->route('home')-
>with('successMsg','You have been successfully added a new fee payment');
    }
}
```

Other features

- One can view the total amount of fees paid by all students
- One can view all fees payment that have been made by a specific student via a textbox search.

Deployment

You must deploy your application on a live server accessible from anywhere: There are free web hosting services online or you can buy at 1k and share using sub-domains.

Marking

Upload your completed word document (this one) including the link pointing to where you host your work.

Put your Link here:

`https://git.heroku.com/api102730.git`

Put your repository link here:

--End--