TD 4
Application « THOLDI »

Objectif	Savoir développer au sein du Framework LARAVEL Intégrez un logiciel tiers (dépendance). Réalisez des graphiques
Contexte d'apprentissage	L'entreprise THOLDI propose un module de gestion de réservations de containers, un module de gestion de devis et un module statistique au travers d'une application web.
Domaine d'activité	B2.1 SLAM Concevoir et développer une solution applicative
Ressource(s)	http://laravel.com https://developers.google.com/chart?hl=fr

I. Présentation

A. Google Chart

L'API *Google Charts* permet de générer des graphiques. L'accès à l'API est réalisé en javascript. La documentation et les exemples fournis permettent une mise en œuvre rapide.

Dans ce TD, les données du model sont injectées dans la vue au format json. Une autre approche consiste à écrire l'ensemble des instructions en javascript et d'interroger l'application à l'instar d'un web service.

II. IMPLEMENTATION D'UN MODULE STATISTIQUE

Dans cette partie, il s'agit d'ajouter la partie relative à la consultation d'un module statistique.

1. Routage

✍ Travail à faire ✍	
1	Dans le fichier *routes\web*, définissez un groupe de route, nommée *tableauDeBord* associées au module statistique, ainsi qu'une route permettant d'accéder au graphique de consultation du nombre de reservations par mois.

```
Route::group(['prefix' => 'tableauDeBord'], function () {
    Route::get('nombreDeReservationParMois',
[TableauDeBordController::class, 'nombreDeReservationParMois'])
            ->name('r-nombreDeReservationParMois');
```

2. Controller et Méthode de controller

✍ Travail à faire ✍	
2	Depuis un terminal, créez le contrôleur *TableauDeBordController* via la (ou les) commande(s) *artisan* ci-après

```
./artisan make:controller TableauDeBordController
```

✍ Travail à faire ✍	
3	Dans la classe *TableauDeBordController*, ajoutez une directive permettant au à la classe *TableauDeBordController* d'accéder à la classe *Propel*.

```
use Propel\Runtime\Propel;
```

Pour réaliser des requêtes complexes ou des requêtes qui ne permettent de créer d'objets (on parle d'hydradation d'objets), l'utilisation de la classe *Propel* est à privilégier.
Cf. https://propelorm.org/documentation/03-basic-crud.html#using-custom-sql

✍ Travail à faire ✍	
4	Dans le contrôleur *app/http/Controllers/TableauDeBordController.php*, ajoutez la méthode *nombreDeReservationParMois* comme ci-dessous. Les éléments surlignés sont explicités ci-après

```
public function nombreDeReservationParMois(Request $request) {
     $utilisateur = $request->session()->get('utilisateur');
     $codeUtilisateur = $utilisateur->getCodeutilisateur();
     $pdo =
Propel::getWriteConnection(\App\Http\Model\Map\ReservationTableMap::DATAB
ASE_NAME);
     $sql = "select COUNT(DATE_FORMAT(`dateDebutReservation`, '%m-%Y'))
as nbReservationParMoisPourUnUtilisateur,
DATE_FORMAT(`dateDebutReservation`, '%m-%Y') as moisAnneeDeReservation
from reservation rs
join  utilisateur u on u.codeUtilisateur = rs.codeUtilisateur
where u.codeUtilisateur = :codeUtilisateur
group by DATE_FORMAT(`dateDebutReservation`, '%m-%Y')";

   $pdoStatement = $pdo->prepare($sql);
   $pdoStatement->execute(array(':codeUtilisateur' => $codeUtilisateur));
    $data = $pdoStatement->fetchAll(\PDO::FETCH_ASSOC);

    return view('tableauDeBord.nombreDeReservationParMois',
                ['dataJson'=> json_encode($data)]
        );
    }
```

La fonction *DATE_FORMAT* retourne une date à partir d'un format transmis en paramètre.
Cf. https://sql.sh/fonctions/date_format

La constante *PDO::FETCH_ASSOC* permet de préciser le format de la valeur de retour : tableau associatif, tableau indexé, modèle objet etc.)

La fonction *json_encode* permet d'encoder les données sous la forme d'une chaine de caractères *json*. Il s'agit d'un format d'échange à l'instar d'XML.
Cf. https://www.json.org/json-fr.html

a) Menu principal

✍ Travail à faire ✍	
5	Compléter dans le fichier *\resssources\views\layouts\default.blade.php*, le lien hypertexte d'accès au graphique du nombre de réservation par mois du client authentifié.

```
<a class="dropdown-item" href="{{ route('r-
nombreDeReservationParMois')}}">Nombre de réservations (par
mois)</a>
```

b) Consultation d'un graphique

✍ Travail à faire ✍	
6	Depuis un terminal, créez le répertoire *resources\views\tableauDeBord*.

```
cd /var/wwww/html/tholdi-resa
mkdir resources/views/tableauDeBord
```

✍ Travail à faire ✍	
7	Depuis le répertoire *tableauDeBord*, créez un fichier de vue nommé *nombreDeReservationParMois.blade.php* et complétez-le comme ci-dessous.

L'interface graphique ci-dessous intègre le code javascript fourni en exemple sur le site de Google Chart. A l'aide de la documentation, il est possible de personnaliser les graphiques.

En résumé, le code javascript réaliser les traitements suivants :
- Charger l'API google Chart
- De définir une méthode déclenchée quand l'API est chargée
- D'initialiser un objet BarChart (graphique en barre) et de le dessiner dans une zone html défini (dans l'exemple, il s'agit de la balise div dont l'ID est chart_div)

La méthode réalise les traitements suivants :

- Création d'un objet DataTable (un objet datable contient l'ensemble des données à utiliser pour la construction d'un objet graphique Google Chart) , un ensemble de ligne et de colonnes.
- Ajout des lignes de données à l'objet DataTable
- Personnalisation de l'objet graphique via un tableau d'options
- Initialisation et affichage du graphique à partir des données et du tableau d'options

Les données injectées par le controleur à la vue est au format json. Une directive Blade est utilisée pour insérer dans le script javascript la chaine au format json contenant les données statistiques à utiliser.
« Le JavaScript Object Notation (JSON) est un format standard utilisé pour représenter des données structurées de façon semblable aux objets Javascript. Il est habituellement utilisé pour structurer et transmettre des données sur des sites web (par exemple, envoyer des données depuis un serveur vers un client afin de les afficher sur une page web ou vice versa) ».
Cf. (https://developer.mozilla.org/fr/docs/Learn/JavaScript/Objects/JSON)

```
@extends('layouts.default')

@section('title')

<h3>
     Nombre de réservations (réservations en cours, validées ou
effectuées) par mois
</h3>

@endsection

@section('content')

<div class="row">
    <div class="col-6">

        <!--Load the AJAX API-->
        <script type="text/javascript"
src="https://www.gstatic.com/charts/loader.js"></script>
        <script type="text/javascript">

            // Load the Visualization API and the corechart package.
            google.charts.load('current', {'packages': ['corechart']});

            // Set a callback to run when the Google Visualization API is
loaded.
            google.charts.setOnLoadCallback(drawChart);

            // Callback that creates and populates a data table,
            // instantiates the pie chart, passes in the data and
            // draws it.
            function drawChart() {

                // Create the data table.
                var dataTable = new google.visualization.DataTable();
                dataTable.addColumn('string', 'Mois Année');
                dataTable.addColumn('number', 'Nombre de réservation');
                        var data = {!! $dataJson !!}
```

```
                data.forEach(function (element) {

                    dataTable.addRows([
                        [element["moisAnneeDeReservation"],

element["nbReservationParMoisPourUnUtilisateur"]]
                    ]);
                });

                // Set chart options
                var options = {'title': 'Répartition des réservations par
mois au cours de l\'année courante',
                    'width': 900,
                    'height': 500

                };

                // Instantiate and draw our chart, passing in some
options.
                var chart = new
google.visualization.BarChart(document.getElementById('chart_div'));
                chart.draw(dataTable, options);
            }
        </script>

     <!--Div that will hold the pie chart-->
        <div id="chart_div">        </div>
    </div>

</div>
</div>

@endsection
```

3. Test

✍ Travail à faire ✍	
8	Démarrez l'application et consultez le graphique depuis la section tableau de bord.