Faculty of Engineering, Computing & Science

ngPigeon Research Project

Pass Task 8: Evaluating Custom Framework

# Overview

The “ngPigeon” project was initiated back in 2017, which aims to ease web-app developers in their web application development process, to enhance developers’ web application development experience and to optimize their productivity. This project focuses on developing various modules or components required for typical data-driven web application development. As to-date, two modules has been developed namely pigeon-table and pigeon-chart.

|  |  |
| --- | --- |
| **Purpose:** | To experience and evaluate nano-framework develop by ngPigeon Projects Team. |
| **Task:** | Construct a few tables and charts using ngPigeon modules with imDB dataset provided. |
| **Time:** | Before the end of Week 12 |
| **Resources:** | * <https://github.com/ngPigeon/pigeon-table> * <https://github.com/ngPigeon/pigeon-chart> * Pigeon-table paper: <https://ieeexplore.ieee.org/document/8270395/> * Pigeon-Chart paper <https://ieeexplore.ieee.org/document/8644793> |
|  |  |

Submission Details

You must submit the following files to Canvas:

* Your project folder.
* Fill in the developer experience feedback form after completing the tasks below.

Your input will be mainly used for research & development purpose and will not affect the grade/marks you obtained in the unit of study.



# Instructions

**Data Preparation**

1. Download imdb data from link provided “\*.sql”
2. Launch XAMPP, start Apache & MySQL
3. Go to phpmyadmin and then create a database named “imdb”
4. Import imdb data using .sql file downloaded earlier

# pigeon-table

**Setup**

1. Download pigeon-table package from <https://github.com/ngPigeon/pigeon-table>
2. Extract and Unzip into htdocs folder.
3. Configure database credential in the file named “pigeon-core/configdb.php”
4. Enter hostname, username, password and database as ”imdb”

**Usage**

1. Open the file example.php, inject pigeon-table in html tag: ng-app=”pigeon-table”
2. Within the head tag, include all required dependencies from the file “pigeon-table/php/includes.php”:

<!DOCTYPE html>

<html lang="en" **data-ng-app="pigeon-table"**>

<head>

<title>Demo: pigeon-table</title>

**<?php**

**include "pigeon-table/php/includes.php"**

**?>**

</head>

<body>

**<pigeon-table query=”SELECT \* FROM actor”></pigeon-table>**

</body>

</html>

1. Within the <body> tag, use <pigeon-table></pigeon-table> to render a table contain all actors’ information.
   1. Try to search any movie of your choice (exact match), perform exclusion, sorting, and paginate the records.
2. Pigeon-table can be configure with the following attributes:
   1. **Query**: SQL select statement
   2. **Control**: true (default) or false
   3. **Editable**:true or false (default)
3. Try add in **control** attribute and **set it to false**, refresh the page. Notice that some fields on top of the table are hidden.
4. Set the control back to true, now add **editable** attribute and **set it to true**.
   1. Refresh the page and notice that Insert button appear on top of the table while edit and delete button appear next to each row of record.
   2. Try to add in your information into Actor table, Edit it, and Delete It.
5. Try to change the SQL statement to something else i.e.:
   1. **Table joining**:

**SELECT** m.title 'MovieTitle', m.relyear 'YearRelease',

r.ratingcode 'RatingCode', r.shortdesc 'Description',

m.tmdb\_votes 'VoteCount'

**FROM** movie m

**INNER JOIN** rating r

**ON** m.ratingcode=r.ratingcode

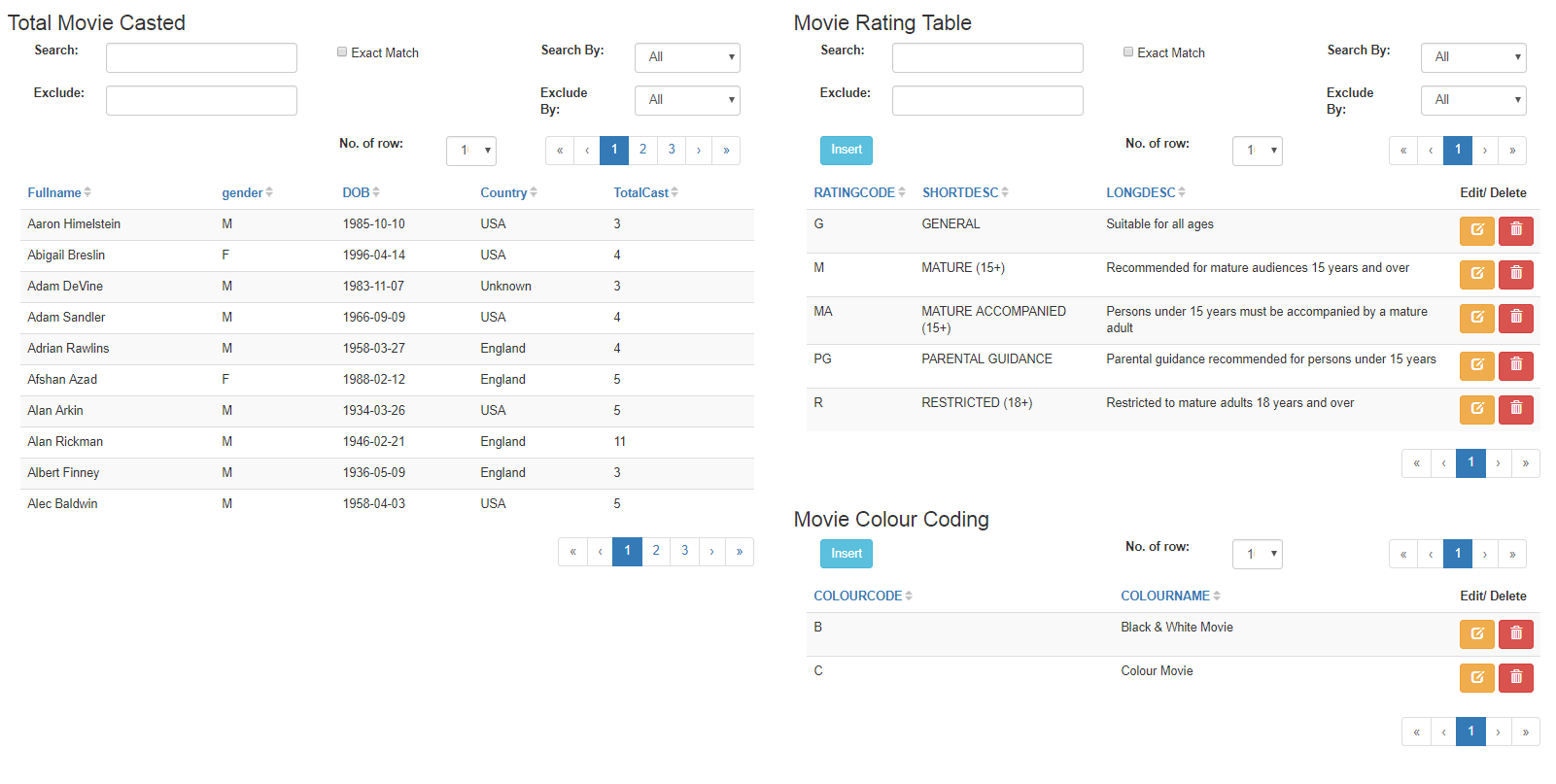
* 1. **Aggregation Result**:

**SELECT** birthcountry, gender, count(\*) 'TotalActor'

**FROM** actor

**GROUP BY** birthcountry, gender

1. You may wrap <pigeon-table> using bootstraps grid system to control its width. On top of that, you can create multiple tables within a webpage. Now try to create the following dashboard with a few table using pigeon-table & bootstraps grid system



**Tip**: **SELECT** a.fullname 'Fullname', a.gender, a.birthdate 'DOB',

a.birthcountry 'Country', count(c.castid) 'TotalCast'

**FROM** actor a **INNER JOIN** casting c

**ON** a.actorno = c.actorno **GROUP BY** a.fullname

As part of continuous effort in improving the framework, the research team welcome regular feedback from web developers. Do tell us what you think about this framework by fill in a survey form provided at the end of this handout **after completing pigeon-chart task** below.

# pigeon-chart

**Setup**

1. Download pigeon-table package from <https://github.com/ngPigeon/pigeon-chart>
2. Extract and Unzip into htdocs folder.
3. Configure database credential in the file named “pigeon-core/configdb.php”
4. Enter hostname, username, password and database as ”imdb”

**Usage**

1. Open the file example.php, inject pigeon-chart in html tag: ng-app=”pigeon-chart”
2. Within the head tag, include all required dependencies from the file “pigeon-chart/php/includes.php”:

<!DOCTYPE html>

<html lang="en" **data-ng-app="pigeon-chart"**>

<head>

<title>Demo: pigeon-table</title>

**<?php**

**include "pigeon-chart/php/includes.php"**

**?>**

</head>

<body>

<!-- create single series line chart -->

**<pigeon-chart query="**SELECT relyear, count(relyear)

FROM movie

GROUP BY relyear**"**

**type="**line**"></pigeon-chart>**

</body>

</html>

1. Within the <body> tag, use <pigeon-chart></pigeon-chart> to render desire table.
2. Pigeon-chart can be configure with the following attributes (add one by one and observe the changes):
   1. **Type**: column [ line/bar/spline/area ]
   2. **Title**: Total Number of Movies
   3. **Subtitle**: Since 1953 – 2016
   4. **Axisx-title**: Year
   5. **Axisy-title**: Count
   6. **Data-data-label**: true (enable actual value for each data point) [false]
   7. **Show-legend**: true [false]
3. Next try to duplicate the code above and **create multi-series line chart** with multiple aggregated values. Look at the min, average and max movie runtime over the years

**SELECT** relyear, min(runtime) 'Min. Duration',

avg(runtime) 'Avg. Duration',

max(runtime) 'Max. Duration'

**FROM** movie

**GROUP BY** relyear

1. Now try to create a pie chart to show number of movie by rating:

<pigeon-chart **query**="SELECT ratingcode, count(ratingcode)

FROM movie

GROUP BY ratingcode"

**title**="Total Number of Movie by Rating"

**type**="pie"

**axisy-title**="Movie Count"

**show-legend**="true"

**data-data-label**="true">

</pigeon-chart>

1. The SQL statement will determine how the chart to be render. Below are some of the valid SQL pattern:

|  |  |
| --- | --- |
| **Pattern** | **Description** |
| SELECT <x-axis>, <y-axis>  FROM <table-name> | Basic Single Series Chart  (line / column / bar / spline / area / pie) |
| SELECT <x-axis>, <y-axis-s1>, <y-axis-s2>, …  FROM <table-name> | Basic Multi-Series Chart  (line / column / bar / spline / area) |
| SELECT <x-axis-g1>, <x-axis-g2>, …, AGGR(y-axis)  FROM <table-name>  GROUP BY <x-axis-g1>, <x-axis-g2>, …, | Multi-Level (Group) Single Series Chart  (line / column / bar / spline / area) |
| SELECT <x-axis-g1>, <x-axis-g2>, …,  AGGR(y-axis-s1), AGGR(y-axis-s2), …  FROM <table-name>  GROUP BY <x-axis-g1>, <x-axis-g2>, …, | Multi-Level (Group) Multi Series Chart  (line / column / bar / spline / area) |
| SELECT <category>, <value>  FROM <table> | Pie Chart |

1. Similar to pigeon-table, pigeon-chart support table join query as well.
2. You may wrap <pigeon-chart> using bootstraps grid system to control its width. On top of that, you can create multiple chart within a webpage.

