Lab 10: Database programming with ASP.net C# to make dash board

จุดประสงค์

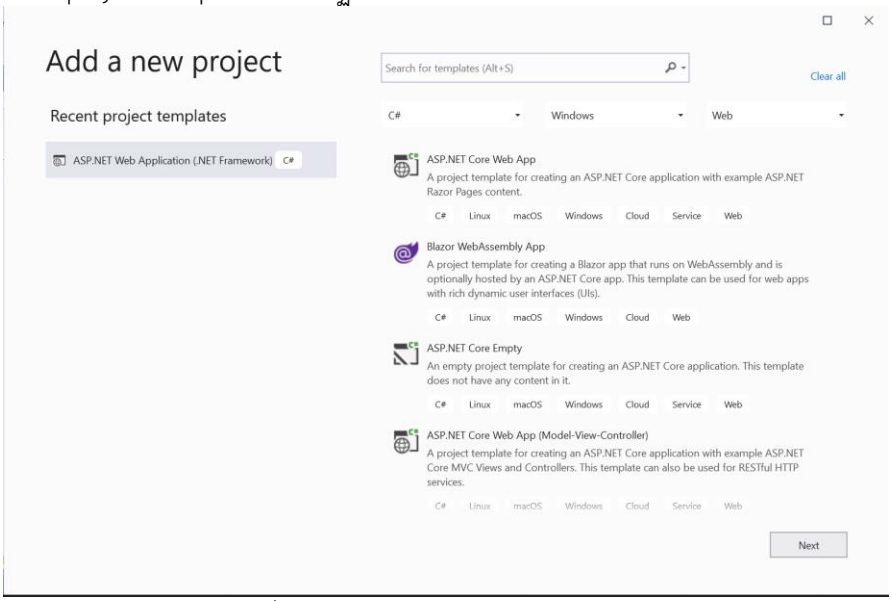
ศึกษาและเรียนรู้ Database programming ด้วย ASP.net C# ร่วมกับ JavaScript

ความต้องการ

1. MS SQL Server Express or above link download
2. SQL Server Management Studio version 18 or above editor link download
3. Microsoft Visual Studio community 2019 or above link download
4. Dashboard template link download
5. SQL file with structure and data link download

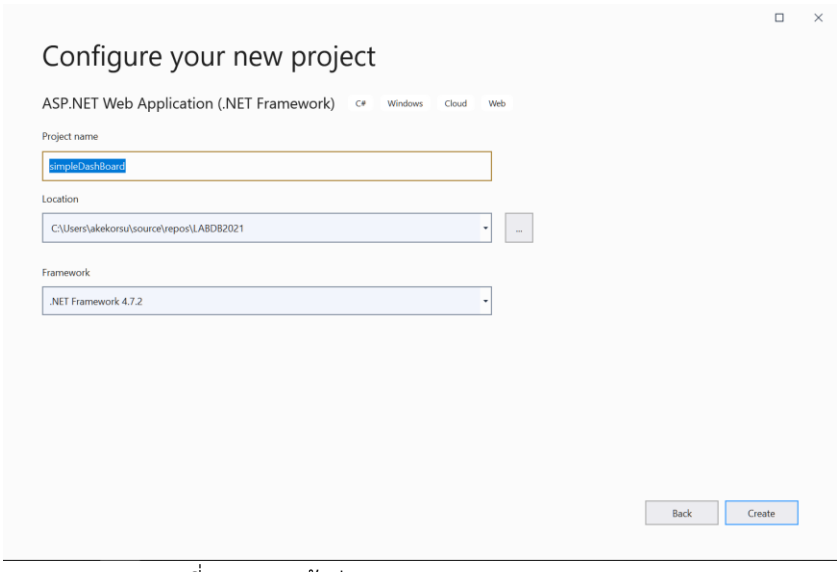
การทดลอง

1. เปิดโปรแกรม visual studio 2019 แล้วเปิด Solution ล่าสุดที่ได้สร้างไว้ชื่อ “LABDB20121.sln”
2. แล้วไปยัง Solution Explorer แล้วทำการสร้าง project ใหม่ ด้วยการ right click ที่ Solution ‘LABDB2021’ แล้วเลือก Add -> New Project...
3. แล้วระบบจะแสดงหน้าต่าง “Add a new project” ดังรูปที่ 1 ซึ่งหลังจากที่เคยสร้าง Project ขึ้นมาแล้ว จะมี “Recent project templates” ทำให้สะดวกไม่จำเป็นต้องค้นหา ภาษา แพลตฟอร์ม ที่จะใช้ในการพัฒนา ในที่นี้เราก็ทำการ click เลือก project template ที่ปรากฏใน recent นั้น



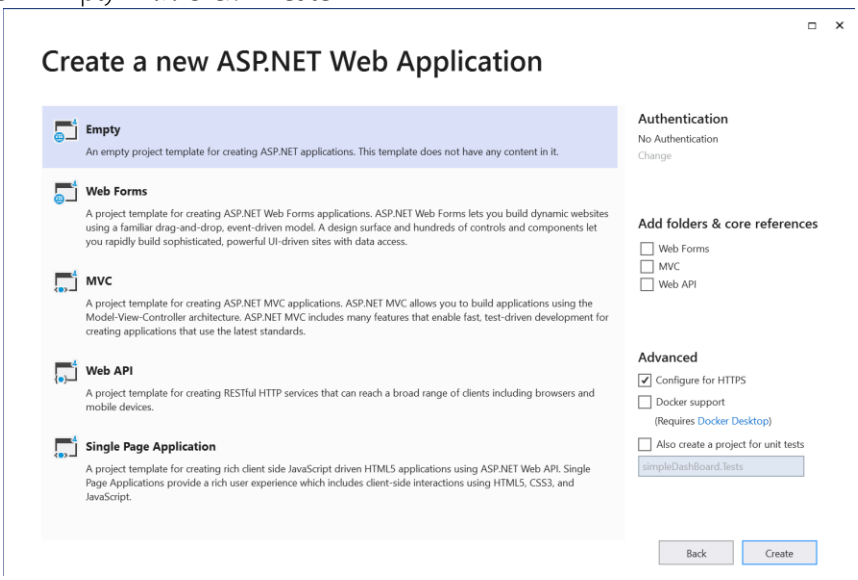
รูปที่ 1 แสดงหน้าต่าง Add a new project

4. เมื่อทำการเลือก template project เรียบร้อยแล้ว จะปรากฏหน้าต่าง “Configure your new project” ดังรูปที่ 2 ในส่วนของ “Project name” ให้ทำการตั้งชื่อว่า “simpleDashBoard” แล้วทำการ click “Create”



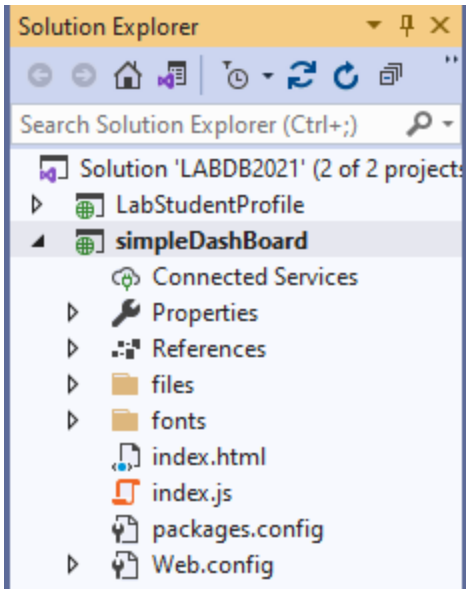
รูปที่ 2 แสดงหน้าต่าง Configure your new project

5. เมื่อทำการตั้งชื่อ project เรียบร้อยแล้ว จะปรากฏหน้าต่าง “Create a new ASP.NET Web Application” ดังรูปที่ 3 ให้ทำการเลือก “Empty” แล้ว click “Create”



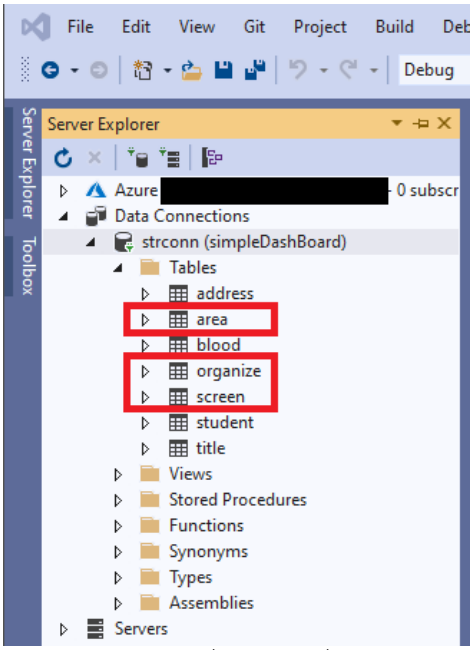
รูปที่ 3 แสดงหน้าต่าง Create a new ASP.NET Web Application

6. แล้วระบบจะนำไปสู่หน้าทำงานหลัก ซึ่งเมื่อมองดู “Solution Explorer” จะเห็นได้ว่ามี 2 project ภายใน solution เดียวกัน แต่ชื่อ project สองอันนั้นจะมีลักษณะที่แตกต่างกัน อันหนึ่งตัวหนาอันหนึ่งตัวบาง ให้ทำการ right click ที่ project “simpleDashBoard” เลือก “Set as Startup Project” ซึ่งจะทำให้ project ใหม่ที่ได้สร้างนั้นกลายเป็นตัวหนังสือหนา และเป็นตัวทำงานหลักกรณีที่เราทำการ run WepApplication
7. หลังจากนั้นให้ Copy การตั้งค่าของไฟล์ Web.config จาก project “LabStudentProfile” มายัง project “simpleDashBoadrd” เพื่อทำการตั้งค่าการเชื่อมต่อฐานข้อมูลอันเดียวกัน
8. แล้วจึง Copy ไฟล์ภายใน Dashboard template ที่ระบุไว้ในส่วนความต้องการลำดับที่ 4. ทั้งหมดมายัง project “simpleDashBoard” แล้วจะได้ Solution Explorer ดังรูปที่ 4



รูปที่ 4 แสดง Solution Explorer หลังจากการ Copy ไฟล์ Dashboard template

9. มาต่อด้วยการจัดเตรียม ฐานข้อมูลที่จะใช้สำหรับสร้าง Dashboard โดยทำการ execute query ไฟล์ที่ระบุไว้ในส่วนความต้องการที่ 5. ซึ่งเมื่อทำการ execute แล้วนั้นจะมีตารางเพิ่มเข้ามาในฐานข้อมูล โดยสามารถที่จะตรวจสอบได้จาก Server Explorer ดังรูปที่ 5 ตารางที่เพิ่มขึ้นมานั้นได้ทำการวงไว้ด้วยกรอบสีแดง



รูปที่ 5 แสดง Server Explorer ภายหลังที่ได้ทำการเพิ่มตารางใหม่สำหรับทำ dashboard

10. โดยจะมี 3 ตารางรายละเอียดดังนี้

area คือ ตารางเขต

column	ความหมาย
id	รหัสเขต
name	ชื่อเขต

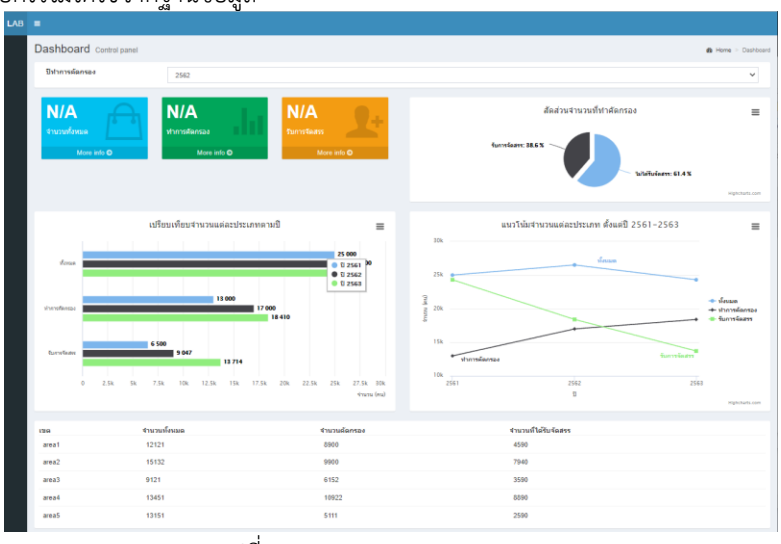
organize คือ ตารางหน่วยงาน

column	ความหมาย
a_id	รหัสเขต
o_id	รหัสหน่วยงาน
o_name	ชื่อหน่วยงาน

screen คือ ตารางการคัดกรอง

column	ความหมาย
year	ปีทำการคัดกรอง
o_id	รหัสหน่วยงาน
total	จำนวนคนทั้งหมด
screen	จำนวนคนที่ได้รับทำการคัดกรอง
provide	จำนวนคนที่ได้รับการจัดสรร

11. หลังจากที่เตรียมพร้อมในทั้ง template และฐานข้อมูลแล้วนั้นให้ทดลอง run Web Application จะได้หน้าเว็ปไซต์เหมือนกันกับในรูปที่ 6 แต่เมื่อสังเกตจะเห็นได้ว่ามีข้อมูลบ้างส่วนยังไม่ได้แสดงและข้อมูลในกราฟ รวมถึงตารางก็เป็นเพียงข้อมูลที่ล็อกไว้ไม่ได้ใช้จากฐานข้อมูล

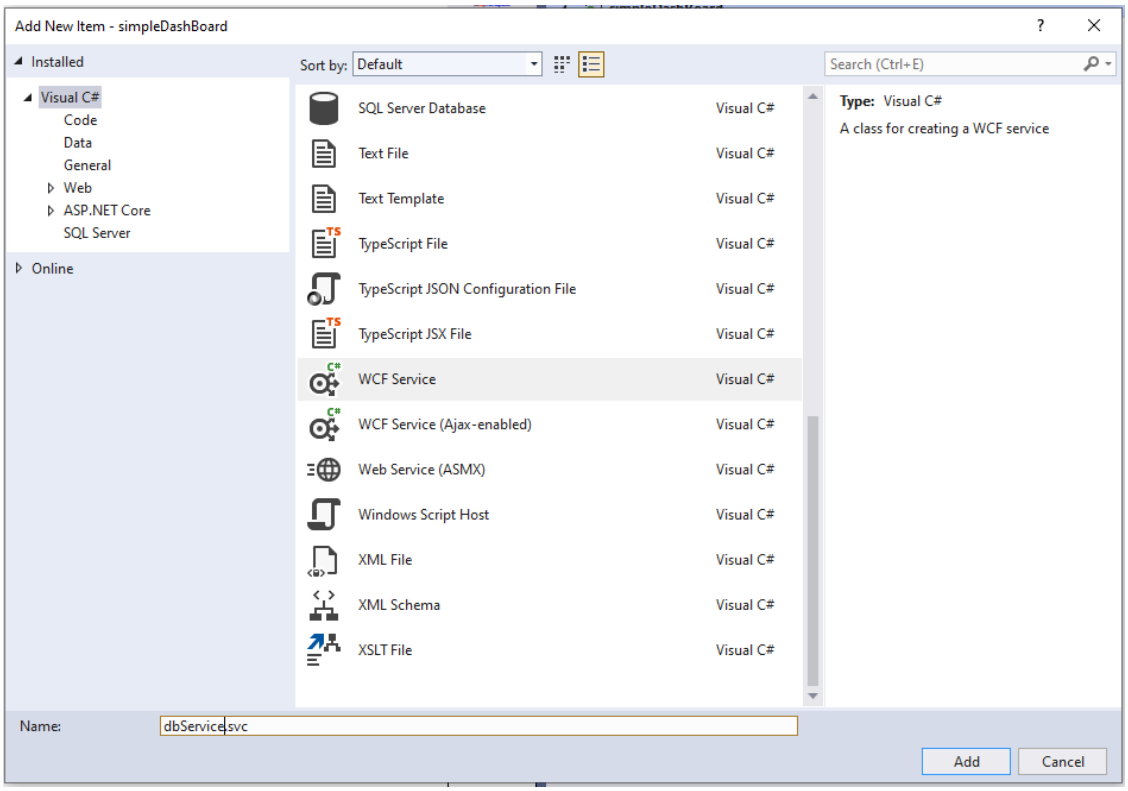


รูปที่ 6 แสดงภาพ dashboard แรก

12. มาต่อที่ visual studio ไปยัง 

พิมพ์ WCF ลงไปแล้วเลือก install more tools and feature แล้วจึงทำการติดตั้งส่วนขยายให้แล้วเสร็จ แล้วจึงไปยัง Solution Explorer ให้ทำการ right click ที่ project “simpleDashBoard” แล้วเลือก Add -> New Item...

13. จะปรากฏหน้าต่าง “Add New Item – simpleDashBoard” ดังรูปที่ 7 ให้ค้นหา “WCF Service” แล้วตั้งชื่อไฟล์ว่า “dbService.svc” แล้วจึง click “Add”



รูปที่ 7 แสดงหน้าต่าง Add New Item สำหรับสร้าง WCF Service

14. right click dbService.svc แล้วเลือก View Markup แล้วแทนที่ code ด้วยกรอบ สีขาวด้านล่าง

```
<%@ ServiceHost Language="C#" Debug="true" Service="simpleDashBoard.dbService"
CodeBehind="dbService.svc.cs"
Factory="System.ServiceModel.Activation.WebServiceHostFactory"%>
```

15. right click project “simpleDashBoard” แล้วเลือก Add -> Class... จะปรากฏหน้าต่าง “Add New Item” ให้ตั้งชื่อว่า “dataContract.cs” แล้ว click “Add”

16. ไปยังไฟล์ dataContract.cs แล้วแทนที่ code ทั้งหมดด้วยกรอบสีขาวด้านล่าง

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

  public class screenData
  {
    public string year;
    public string a_id;
    public string area;
    public string o_id;
    public string o_name;
    public string total;
    public string screen;
    public string provide;
  }
```

17. ไปยังไฟล์ “IdbService.cs” แล้วแทนที่ code ทั้งหมดด้วยกรอบสีขาวด้านล่าง โดยเมื่อแทนที่เสร็จแล้ว ต้องดำเนินการ QuickAction แก้จนหมด error

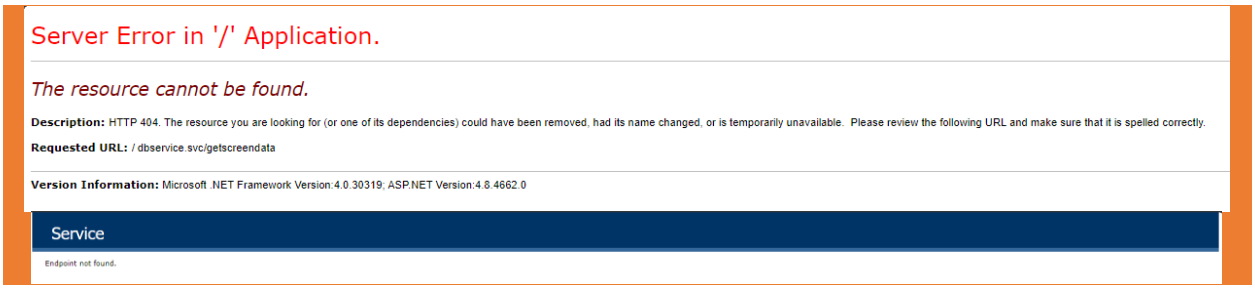
```
using System;
```

```
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;

namespace simpleDashBoard
{
  // NOTE: You can use the "Rename" command on the "Refactor" menu to change the
interface name "IdbService" in both code and config file together.
  [ServiceContract]
  public interface IdbService
  {
    [WebInvoke(Method = "GET", ResponseFormat = WebMessageFormat.Json, UriTemplate =
"getScreenData?year={year}")]
    screenData[] getScreenData(string year );
  }
}
```

18. ไปยังไฟล์ “dbService.svc.cs” แล้วแทนที่ code ทั้งหมดด้วยกรอบสีขาวด้านล่าง โดยเมื่อแทนที่เสร็จแล้ว ต้องดำเนินการ QuickAction แก้จนหมด error แล้วทดลอง run Web Application แล้วแก้ไข URL ใน Browser ไปยัง https://localhost:yourport/ dbservice.svc/getscreendata แล้ว capture ผลลัพธ์ที่ได้ลงในกรอบสีส้ม

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;
using System.Web.Configuration;

namespace simpleDashBoard
{
  // NOTE: You can use the "Rename" command on the "Refactor" menu to change the class
name "dbService" in code, svc and config file together.
  // NOTE: In order to launch WCF Test Client for testing this service, please select
dbService.svc or dbService.svc.cs at the Solution Explorer and start debugging.
  public class dbService : IdbService
  {
    public screenData[] getScreenData(string year)
    {
      List<screenData> retVal = new List<screenData>();
      SqlConnection conn = new
SqlConnection(WebConfigurationManager.ConnectionStrings["strconn"].ConnectionString);
      SqlCommand cmd = new SqlCommand(
        "SELECT S.[year], O.a_id, A.name AS area, O.o_id, O.o_name, S.total, S.screen,
S.provide "+
        "FROM screen S "+
        "JOIN organize O ON  O.o_id = S.o_id "+
        "JOIN area A ON A.id = O.a_id  "
      , conn);
      conn.Open();

      SqlDataReader rdr = cmd.ExecuteReader();

      while (rdr.Read())
```

```
      {
        screenData a = new screenData();
        a.year = rdr[0].ToString();
        a.a_id = rdr[1].ToString();
        a.area = rdr[2].ToString();
        a.o_id = rdr[3].ToString();
        a.o_name = rdr[4].ToString();
        a.total = rdr[5].ToString();
        a.screen = rdr[6].ToString();
        a.provide = rdr[7].ToString();

        retVal.Add(a);
      }

      rdr.Close();
      conn.Close();

      return retVal.ToArray();
    }
  }
}
```

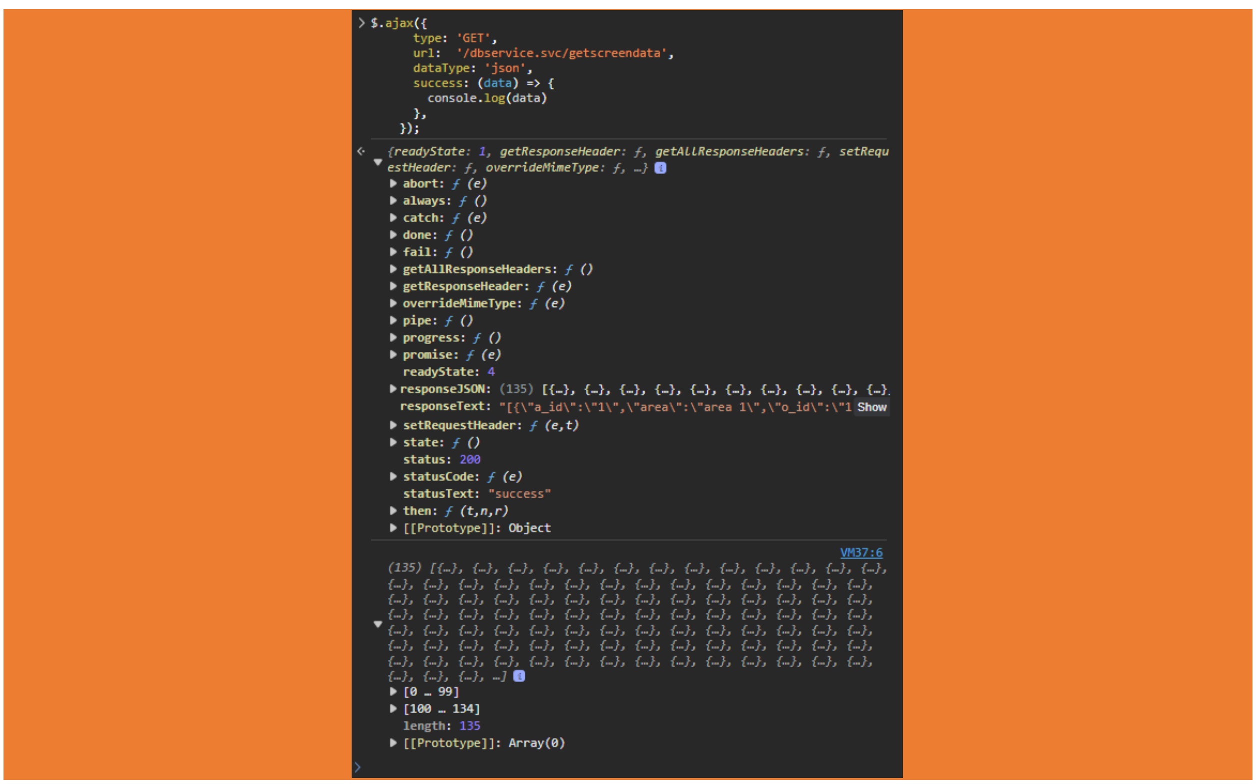


19. แล้วแก้ไข URL ใน Browser ไปยัง index.html แล้วทดลองกด F12 จนปรากฏ DevTools ให้ไปยัง Console แล้ว copy code จากกรอบสีขาวลงไป กด Enter แล้ว capture ผลลัพธ์ที่ได้ลงในกรอบสีส้ม

```
$.ajax({
    type: 'GET',
    url:  '/dbservice.svc/getscreendata',
    dataType: 'json',
    success: (data) => {
      console.log(data)
    },
  });
```

19. กลับไปยัง Web.config ทำการเพิ่ม code ภายใน tag “system.webServer” ด้วย code ในกรอบสีขาวด้านล่าง แล้วทดลอง run Web application ใหม่ แล้วทดลองขั้นตอนที่ 19 อีกครั้งแล้ว capture ผลลงในกรอบสีส้มด้านล่าง

```
<httpProtocol>
  <customHeaders>
    <add name="Access-Control-Allow-Origin" value="*" />
    <add name="Access-Control-Allow-Headers" value="*" />
    <add name="Access-Control-Allow-Methods" value="*" />
    <add name="Access-Control-Max-Age" value="1728000" />
  </customHeaders>
</httpProtocol>
```



20. หลังจากที่เตรียม RestAPI service ผ่าน WCF ใน ASP.Net C# แล้วให้ทำการ run Web Application ไว้ตลอดไม่จำเป็นต้องหยุดอีก เพราะเป็นส่วน Backend ของระบบไว้เชื่อมต่อกับฐานข้อมูล และให้ข้อมูลกับผู้ใช้งาน
21. จะมามุ่งเน้นแก้ไขส่วน Frontend โดยใช้ HTML ในการแสดงผล ร่วมกันกับ Javascript ในส่วนตรรกะและการประมวลผล
22. ให้ทำการเปิดไฟล์ index.js ทำการแทรก code ลงไปในบันทัดที่ 201 โดยใช้จากกรอบสีขาวด้านล่าง

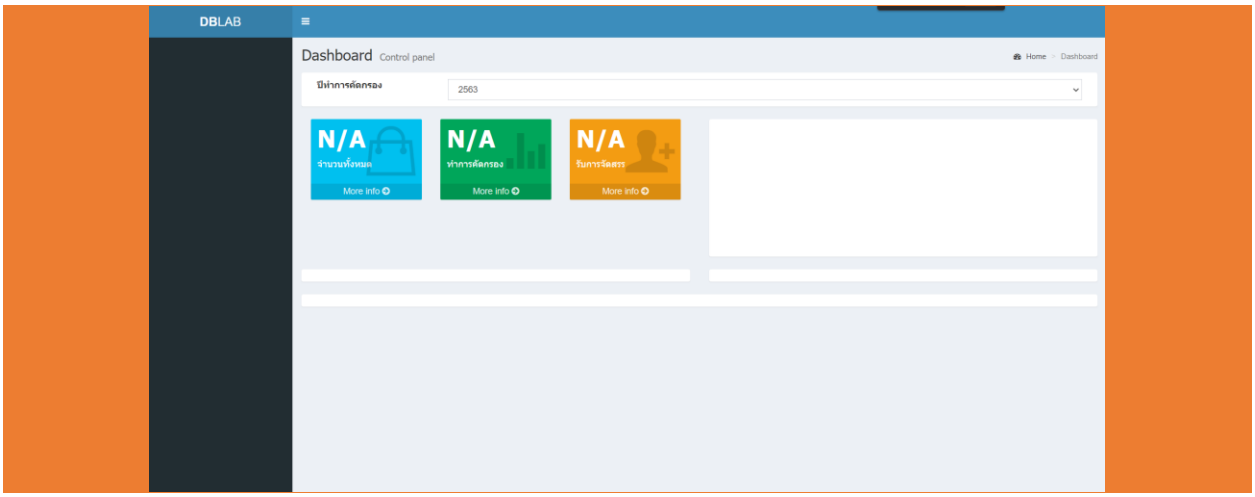
```
var GDATA;
```

23. ทำการแทนที่ code ส่วน **$(document).ready** ทั้งหมดด้วยกรอบสีขาวด้านล่าง

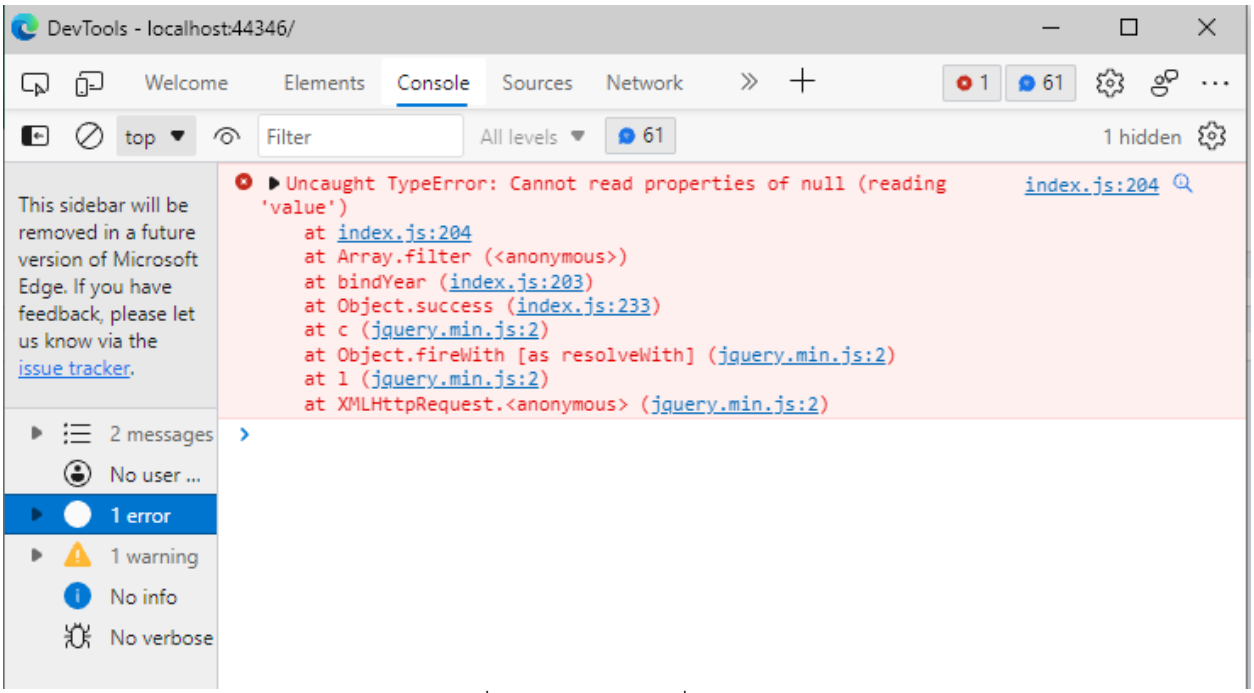
```
$(document).ready(function () {
  $.ajax({
    type: 'GET',
    url: '/dbservice.svc/getscreendata',
    dataType: 'json',
    success: (data) => {
      GDATA = data;

      bindYear();
      createGraph();
      createBar();
    },
  });
});
```

24. ทำการสร้าง function bindYear ขึ้นมาด้านล่างของ function createTable โดยcopy code จากกรอบสีขาวด้านล่าง แต่พอทำการ refresh หน้าเว็ป แล้วทำการ capture ผลที่ได้ลงในกรอบสีส้มด้านล่าง

```
function bindYear() {
  let tmp = GDATA.filter(e => {
    return e.year == element('yearDrpDwn').value
  })

  let sumTotal = 0, sumScreen = 0, sumProvide = 0;

  tmp.forEach(e => {
    sumTotal += parseInt(e.total);
    sumScreen += parseInt(e.screen);
    sumProvide += parseInt(e.provide);
  })

  totalH.textContent = sumTotal;
  screenH.textContent = sumScreen;
  provideH.textContent = sumProvide;

  createPie(sumScreen, sumProvide);
  createTable(tmp);
}
```



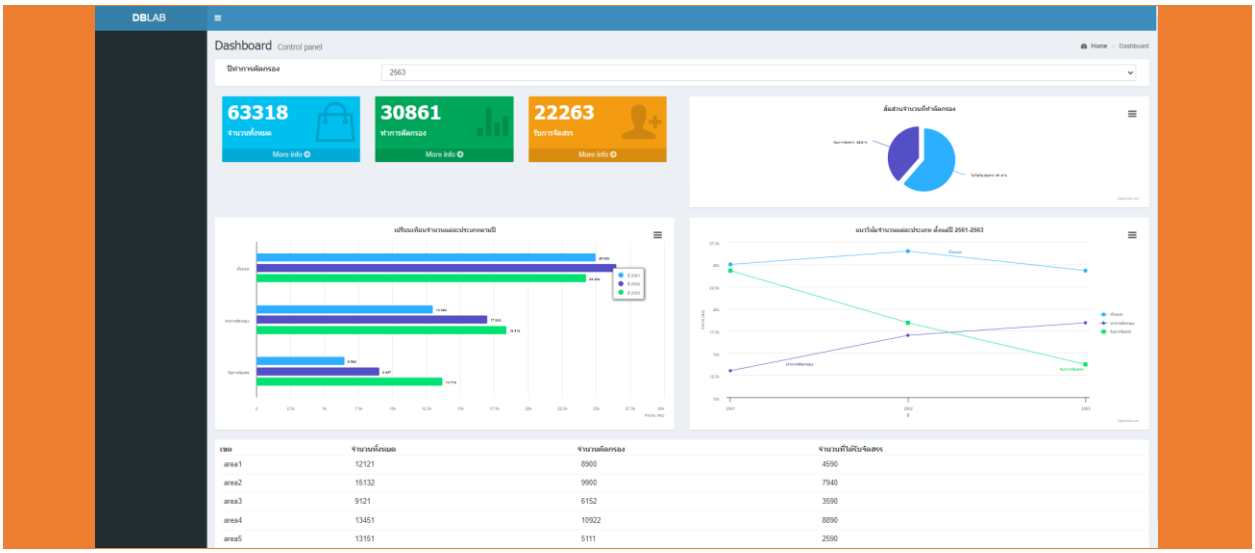
25. ซึ่งเราสามารถดูข้อผิดพลาดของฝั่ง frontend ได้ด้วยการ กด f12 แล้วไป Console จะมี error ขึ้นแสดงดังรูปที่ 8



รูปที่ 8 แสดง error บนฝั่ง frontend

26. เนื่องจากตัว Dropdown ปีการคัดกรองไม่ได้ระบุ id ไว้ให้ทำการแทนที่ code บันทัดที่ 68 ในไฟล์ index.html ด้วยกรอบสีขาวด้านล่าง แล้วทดลอง refresh website อีกครั้ง แล้ว capture ผลที่ได้ลงในกรอบสีส้มด้านล่าง

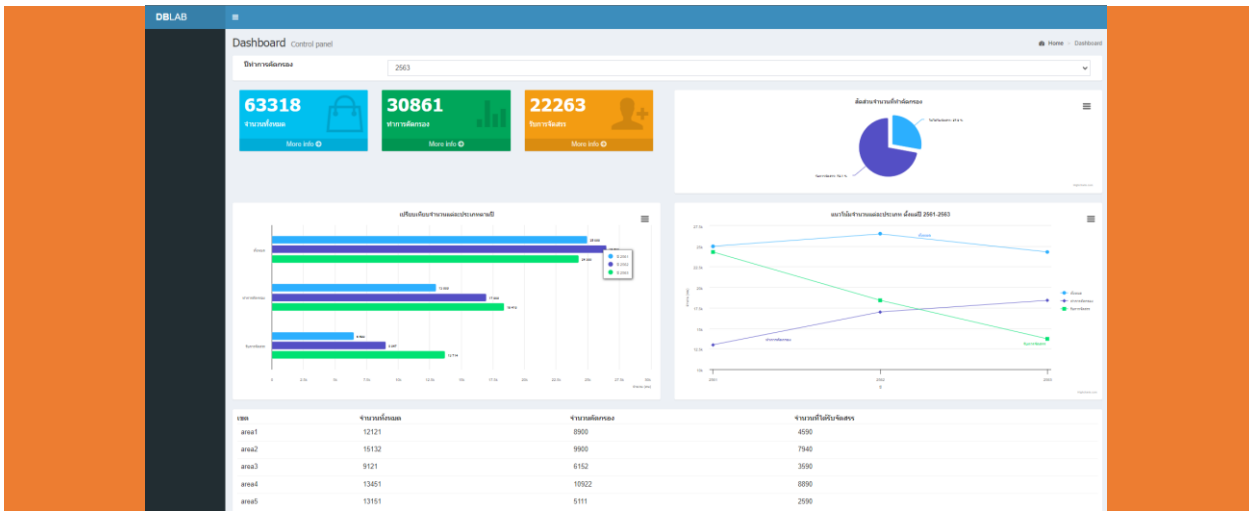
```
<select id="yearDrpDwn" class="form-control">
```



27. ต่อด้วยการแก้ไข function createPie ให้ใช้ข้อมูลจริงด้วยการแทนที่ code ด้วยกรอบสีขาวด้านล่าง แล้วทดลอง refresh website อีกครั้ง แล้ว capture ผลที่ได้ลงในกรอบสีส้มด้านล่าง

```
function createPie(screen, provide) {
  Highcharts.chart('pieContainer', {
    chart: {
      plotBackgroundColor: null,
      plotBorderWidth: null,
      plotShadow: false,
      type: 'pie'
    },
    title: {
      text: 'สัดส่วนจำนวนที่ทำคัดกรอง'
    },
    accessibility: {
      point: {
        valueSuffix: '%'
      }
    },
    plotOptions: {
      pie: {
        allowPointSelect: true,
        cursor: 'pointer',
        dataLabels: {
          enabled: true,
          format: '<b>{point.name}</b>: {point.percentage:.1f} %'
        }
      }
    },
    series: [{
      name: 'Brands',
      colorByPoint: true,
      data: [{
        name: 'ไม่ได้รับจัดสรร',
        y: screen - provide,
        sliced: true,
        selected: true
```

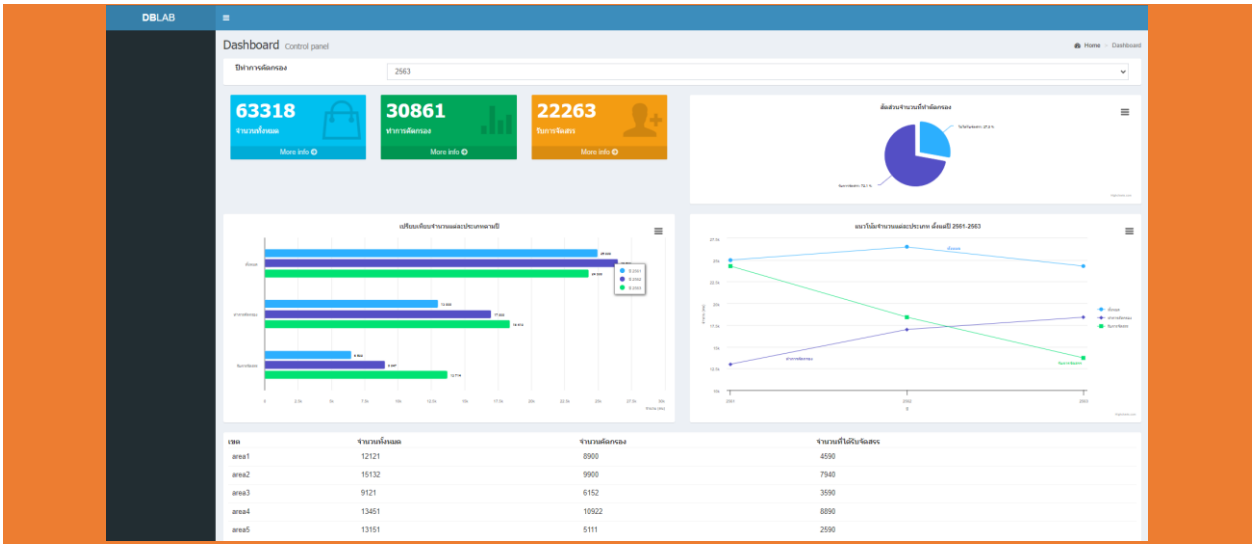
```
      }, {
        name: 'รับการจัดสรร',
        y: provide
      }]
    }]
  });
}
```



28. ต่อด้วยการแก้ไข function **createTable** ให้ใช้ข้อมูลจริงด้วยการแทนที่ code ด้วยกรอบสีขาวด้านล่าง แล้วทดลอง refresh website อีกครั้ง แล้ว capture ผลที่ได้ลงในกรอบสีส้มด้านล่าง

```
function createTable(d, a_id) {
  let thead = element('thead');
  let tbody = element('tbody');

  function createCol(txt, notheader) {
    let c = document.createElement(notheader == undefined? 'th': 'td');
    c.textContent = txt;
    return c
  }
  function createHeader() {
    thead.appendChild(createCol('เขต'));
    thead.appendChild(createCol('จำนวนทั้งหมด'));
    thead.appendChild(createCol('จำนวนคัดกรอง'));
    thead.appendChild(createCol('จำนวนที่ได้รับจัดสรร'));
  }

  function addBody(d) {
    let tr = document.createElement('tr');

    tr.appendChild(createCol(d.name, true))
    tr.appendChild(createCol(d.total, true))
    tr.appendChild(createCol(d.screen, true))
    tr.appendChild(createCol(d.provide, true))

    tbody.appendChild(tr);
```

```
  }

  function createBody(data) {
    let a_id = null, tmp = {};

    data.sort((a, b) => { return a.a_id - b.a_id })

    data.forEach(e => {
      if (e.a_id != a_id) {
        if (a_id != null) addBody(tmp);
        a_id = e.a_id;
        tmp = { a_id: e.a_id, name: e.area, total: parseInt(e.total), screen:
parseInt(e.screen), provide: parseInt(e.provide)  }
      } else {
        tmp.total += parseInt(e.total);
        tmp.screen += parseInt(e.screen);
        tmp.provide += parseInt(e.provide);
      }
    })

    addBody(tmp);
  }

  thead.innerHTML = "";
  tbody.innerHTML = "";

  if (a_id == undefined) {
    createHeader();
    createBody(d);
  }
}
```

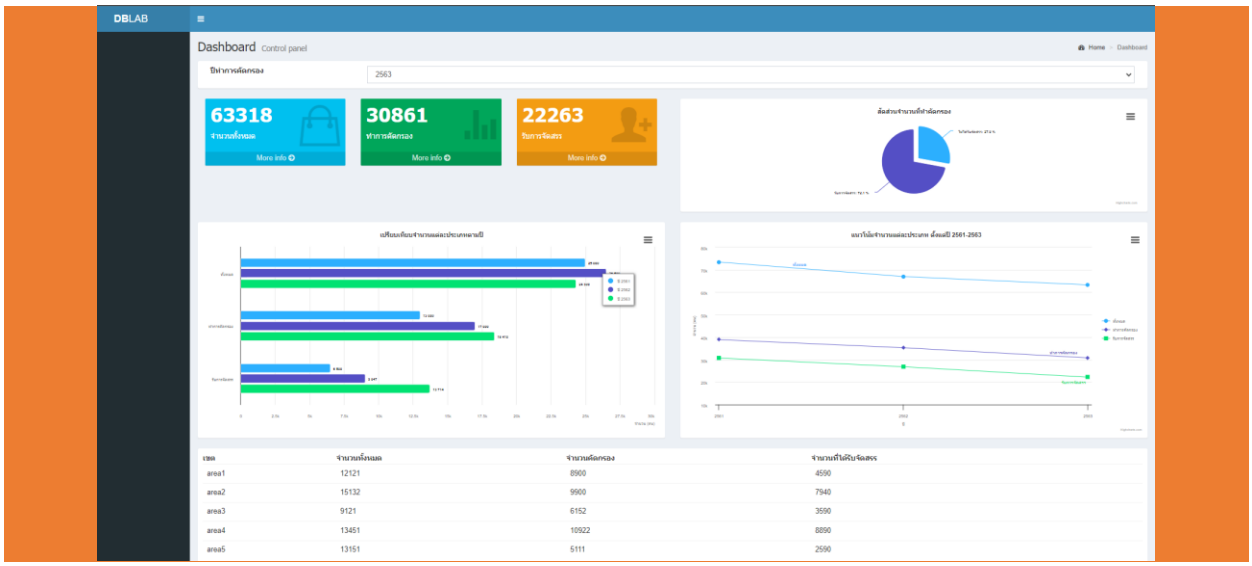


29. ต่อด้วยการแก้ไข function **createGraph** ให้ใช้ข้อมูลจริงด้วยการแทนที่ code ด้วยกรอบสีขาวด้านล่าง แล้วทดลอง refresh website อีกครั้ง แล้ว capture ผลที่ได้ลงในกรอบสีส้มด้านล่าง

```
function createGraph() {
  GDATA.sort((a, b) => {
```

```
  return a.year - b.year;
})

let totals = [], screens = [], provides = [];
let total  = 0, screen  = 0, provide  = 0;
let year = null;

GDATA.forEach(e => {
  if (e.year != year) {
    if (year != null) {
      totals.push(total);
      screens.push(screen);
      provides.push(provide);
    }

    year = e.year;
    total = parseInt(e.total);
    screen = parseInt(e.screen);
    provide = parseInt(e.provide);
  } else {
    total += parseInt(e.total);
    screen += parseInt(e.screen);
    provide += parseInt(e.provide);
  }
});

totals.push(total);
screens.push(screen);
provides.push(provide);

Highcharts.chart('graphContainer', {
  title: {
    text: 'แนวโน้มจำนวนแต่ละประเภท ตั้งแต่ปี 2561-2563'
  },
  yAxis: {
    title: {
      text: 'จำนวน (คน)'
    }
  },
  xAxis: {
    title: {
      text: 'ปี'
    },
    tickInterval: 1
  },
  legend: {
    layout: 'vertical',
    align: 'right',
    verticalAlign: 'middle'
  },
  plotOptions: {
    series: {
      label: {
        connectorAllowed: false
      },
      pointStart: 2561
    }
```

```
    },
    series: [{
      name: 'ทั้งหมด',
      data: totals
    }, {
      name: 'ทำการคัดกรอง',
      data: screens
    }, {
      name: 'รับการจัดสรร',
      data: provides
    }],
    responsive: {
      rules: [{
        condition: {
          maxWidth: 500
        },
        chartOptions: {
          legend: {
            layout: 'horizontal',
            align: 'center',
            verticalAlign: 'bottom'
          }
        }
      }]
    }
  });
}
```

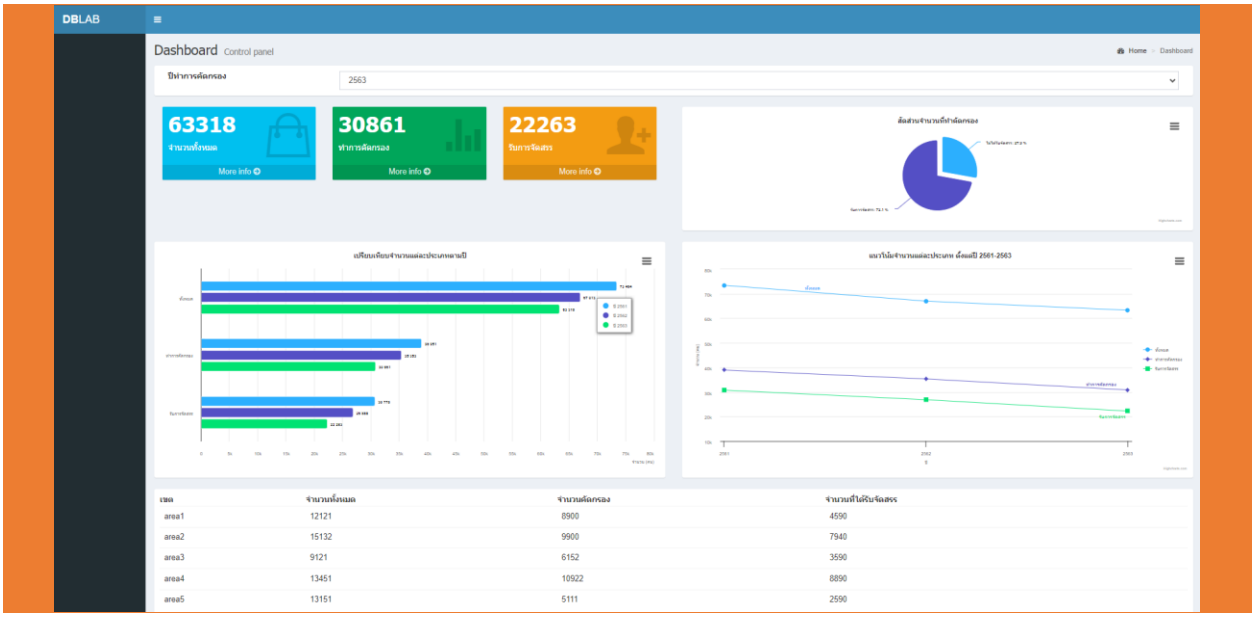


30. ต่อด้วยการแก้ไข function **createBar** ให้ใช้ข้อมูลจริงด้วยการแทนที่ code ด้วยกรอบสีขาวด้านล่าง แล้วทดลอง refresh website อีกครั้ง แล้ว capture ผลที่ได้ลงในกรอบสีส้มด้านล่าง

```
function createBar() {
  GDATA.sort((a, b) => {
    return a.year - b.year;
  })
```

```
let totals = [], screens = [], provides = [];
let total = 0, screen = 0, provide = 0;
let year = null;

GDATA.forEach(e => {
  if (e.year != year) {
    if (year != null) {
      totals.push(total);
      screens.push(screen);
      provides.push(provide);
    }

    year = e.year;
    total = parseInt(e.total);
    screen = parseInt(e.screen);
    provide = parseInt(e.provide);
  } else {
    total += parseInt(e.total);
    screen += parseInt(e.screen);
    provide += parseInt(e.provide);
  }
});

totals.push(total);
screens.push(screen);
provides.push(provide);

Highcharts.chart('barContainer', {
  chart: {
    type: 'bar'
  },
  title: {
    text: 'เปรียบเทียบจำนวนแต่ละประเภทตามปี'
  },
  xAxis: {
    categories: ['ทั้งหมด', 'ทำการคัดกรอง', 'รับการจัดสรร'],
    title: {
      text: null
    }
  },
  yAxis: {
    min: 0,
    title: {
      text: 'จำนวน (คน)',
      align: 'high'
    },
    labels: {
      overflow: 'justify'
    }
  },
  plotOptions: {
    bar: {
      dataLabels: {
        enabled: true
      }
    }
  },
```

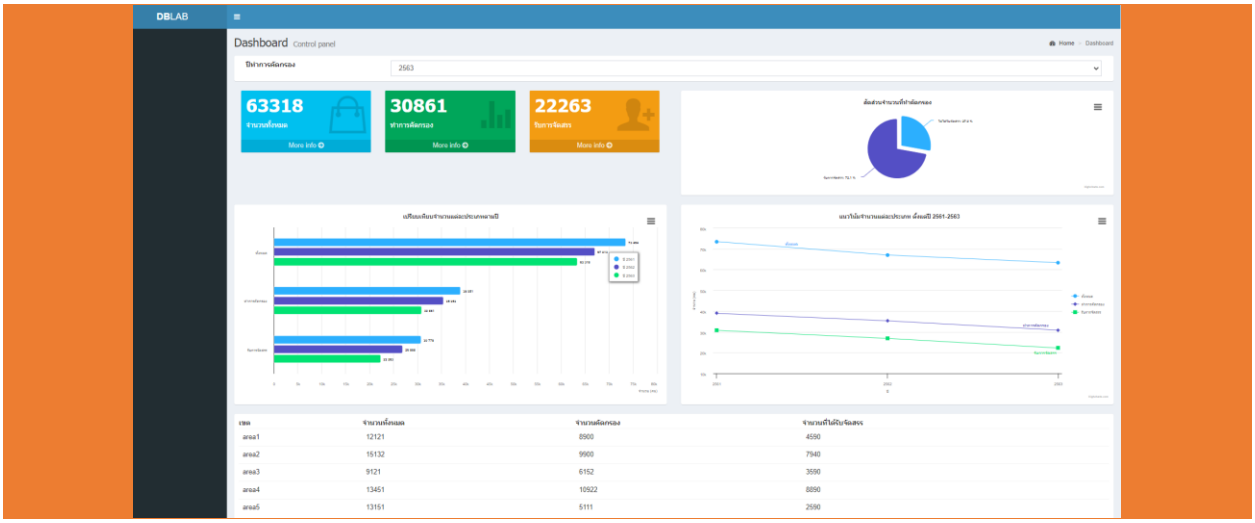
```
    legend: {
      layout: 'vertical',
      align: 'right',
      verticalAlign: 'top',
      x: -40,
      y: 80,
      floating: true,
      borderWidth: 1,
      backgroundColor:
        Highcharts.defaultOptions.legend.backgroundColor || '#FFFFFF',
      shadow: true
    },
    credits: {
      enabled: false
    },
    series: [{
      name: 'ปี 2561',
      data: [totals[0], screens[0], provides[0]]
    }, {
      name: 'ปี 2562',
      data: [totals[1], screens[1], provides[1]]
    }, {
      name: 'ปี 2563',
      data: [totals[2], screens[2], provides[2]]
    },]
  });
}
```



31. มาต่อที่การทำส่วน Filter ในที่นี้จะมีเพียงปีคัดกรองอันเดียวที่เป็น ต้องดำเนินการเพิ่ม event ให้กับการเปลี่ยนค่าของ filter ให้แสดงผลข้อมูลตามปีที่เลือก โดยการแก้ไข `$(document).ready` ด้วยการแทนที่ด้วยกรอบสีขาวด้านล่าง แล้วทดลอง refresh website อีกครั้งปรับเปลี่ยนค่าปีของการคัดกรอง แล้ว capture ผลที่ได้ลงในกรอบสีส้มด้านล่าง

```
$(document).ready(function () {
  element('yearDrpDwn').addEventListener('click', e => {
```

```
    bindYear();
  });

  $.ajax({
    type: 'GET',
    url: '/dbservice.svc/getscreendata',
    dataType: 'json',
    success: (data) => {
      GDATA = data;

      bindYear();
      createGraph();
      createBar();
    },
  });
});
```



32. มาต่อที่ส่วนสุดท้าย Drill down ที่จะทำกับตารางที่แสดงข้อมูล โดยทำการแก้ไข function **createTable** ด้วยการแทนที่ด้วย code ด้านล่างในกรอบสีขาว แล้วทดลอง refresh หน้าเว็ปแล้วกดที่ข้อมูลในตาราง แล้ว capture ผลที่ได้ในตารางลงในกรอบสีส้มด้านล่าง

```
function createTable(d, a_id) {
  let thead = element('thead');
  let tbody = element('tbody');
  let tmpD = d;

  function createCol(txt, notheader) {
    let c = document.createElement(notheader == undefined? 'th': 'td');
    c.textContent = txt;
    return c
  }

  function createHeader(a_id) {
    console.log(a_id)
    thead.appendChild(createCol(a_id == undefined ?'เขต' : 'หน่วยงาน'));
    thead.appendChild(createCol('จำนวนทั้งหมด'));
```

```
    thead.appendChild(createCol('จำนวนคัดกรอง'));
    thead.appendChild(createCol('จำนวนที่ได้รับจัดสรร'));
  }

  function addBody(d, gotEvent) {
    let tr = document.createElement('tr');

    tr.appendChild(createCol(d.name, true))
    tr.appendChild(createCol(d.total, true))
    tr.appendChild(createCol(d.screen, true))
    tr.appendChild(createCol(d.provide, true))

    tr.addEventListener('click', e => {
      createTable(tmpD, d.a_id)
    });

    tbody.appendChild(tr);
  }

  function createBody(data, aid) {
    let a_id = null, tmp = {};

    if (aid == undefined) {
      data.sort((a, b) => { return a.a_id - b.a_id })

      data.forEach(e => {
        if (e.a_id != a_id) {
          if (a_id != null) addBody(tmp);
          a_id = e.a_id;
          tmp = { a_id: e.a_id, name: e.area, total: parseInt(e.total), screen:
parseInt(e.screen), provide: parseInt(e.provide) }
        } else {
          tmp.total += parseInt(e.total);
          tmp.screen += parseInt(e.screen);
          tmp.provide += parseInt(e.provide);
        }
      })

      addBody(tmp, true);
    } else {
      data.filter(e => { return e.a_id == aid})
      data.sort((a, b) => { return a.o_id - b.o_id })

      data.forEach(e => {
        addBody({ a_id: e.a_id, name: e.o_name, total: parseInt(e.total), screen:
parseInt(e.screen), provide: parseInt(e.provide) });
      });
    }
  }

  thead.innerHTML = "";
  tbody.innerHTML = "";

  if (a_id == undefined) {
    createHeader();
    createBody(d);
  } else {
```

```
    createHeader(a_id);
    createBody(d, a_id);
  }
}
```