CIS 219

# COVER PAGE

Introduction to ASP.NET

Dondi Hanson

Chris Ottersen

LAB 3

****

# TABLE OF CONTENTS

[COVER PAGE 1](#_Toc369726981)

[TABLE OF CONTENTS 2](#_Toc369726982)

[NARATIVE 3](#_Toc369726983)

[Set Calculations 3](#_Toc369726984)

[Age Calculator 3](#_Toc369726985)

[Credit Application 3](#_Toc369726986)

[Google Drive hosting 3](#_Toc369726987)

[PROGRAM OUTPUT 4](#_Toc369726988)

[Menu 4](#_Toc369726989)

[NameSort 5](#_Toc369726990)

[How Old?! 7](#_Toc369726991)

[LINK TO RUN PROGRAM ON SERVER 12](#_Toc369726992)

[SOURCE CODE 13](#_Toc369726993)

[Default.aspx 13](#_Toc369726994)

[Default.aspx.cs 14](#_Toc369726995)

[NameSort.aspx 15](#_Toc369726996)

[NameSort.aspx.cs 16](#_Toc369726997)

[SavingsCalculator.aspx 18](#_Toc369726998)

[CreditApplication.aspx.cs 21](#_Toc369726999)

[web.config 25](#_Toc369727000)

# NARATIVE

## Name Sort

This assignment was exceedingly simple. Given this fact you will undoubtedly be shocked when I informed you that I managed to make it substantially more difficult in the name of making the program more dynamic. Rather than simply print: “Enter Name 1 [box]<\br>” I decided that it would be better to place these in a table. Having noted that each row of the table is nearly identical, it seemed like poor practice to simply write the same tags 10 times. Thus I decided that the contents of this page should be placed in an ASP table. In theory, this not only removes the repetition but also, in the event that it became necessary in the future, allows for the number of inputs to be changed to whatever number necessary by changing one number.

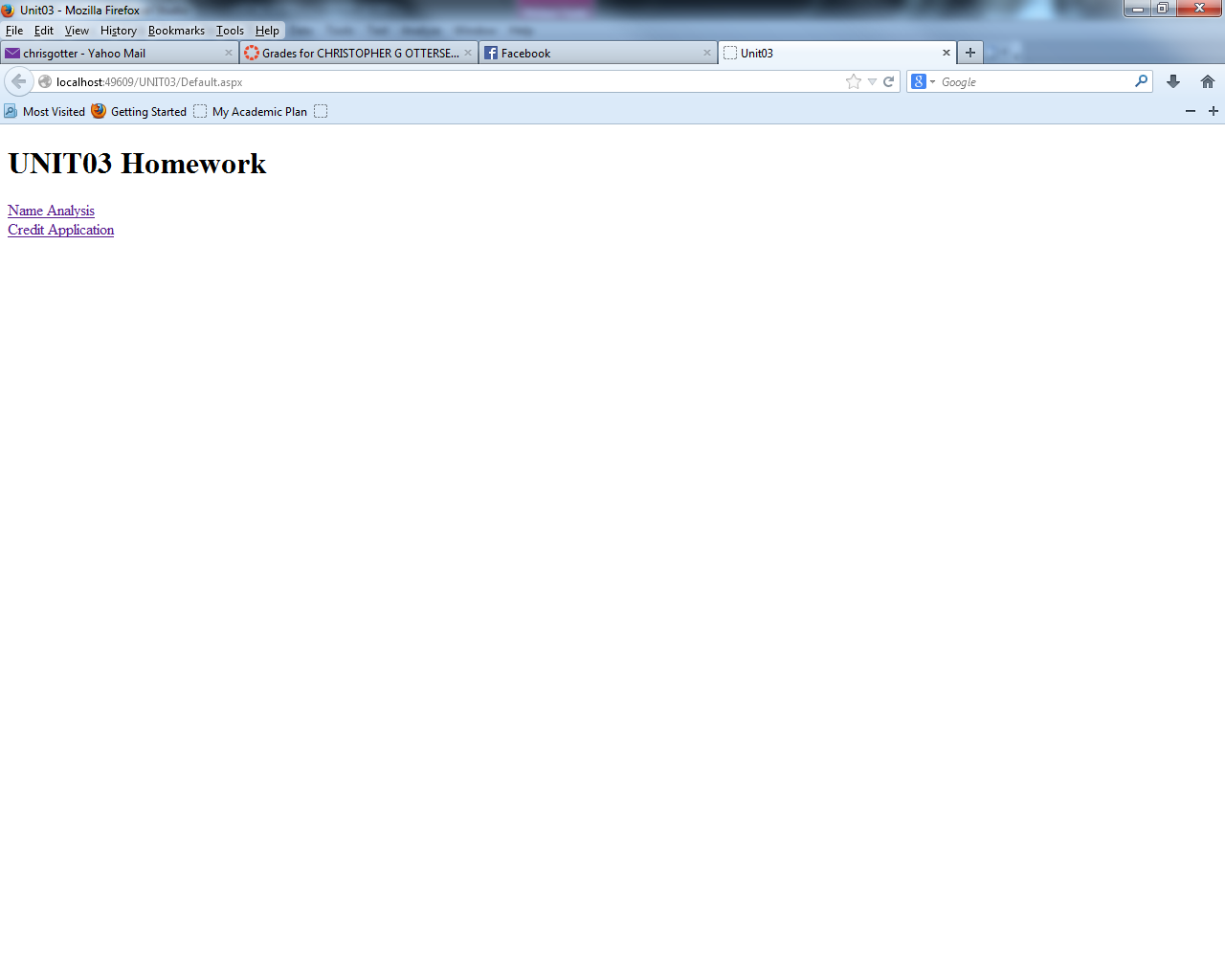
## Savings Calculator

This was my real baby. I spent a substantial amount of time working on this over the past week or so. I misinterpreted the statement in the instructions dictating that we needed to use a table as meaning that we needed to use an ASP table\*. The only reason, that I could think of as to why we would need a dynamically sized table is to display the investment amortization. This is why I elected to display the monthly status. That said, nowhere was it even implied that individual monthly contributions should be editable. I got a great deal out of this lab including experience in the use of dynamically generated controls, asp tables, and auto postback.

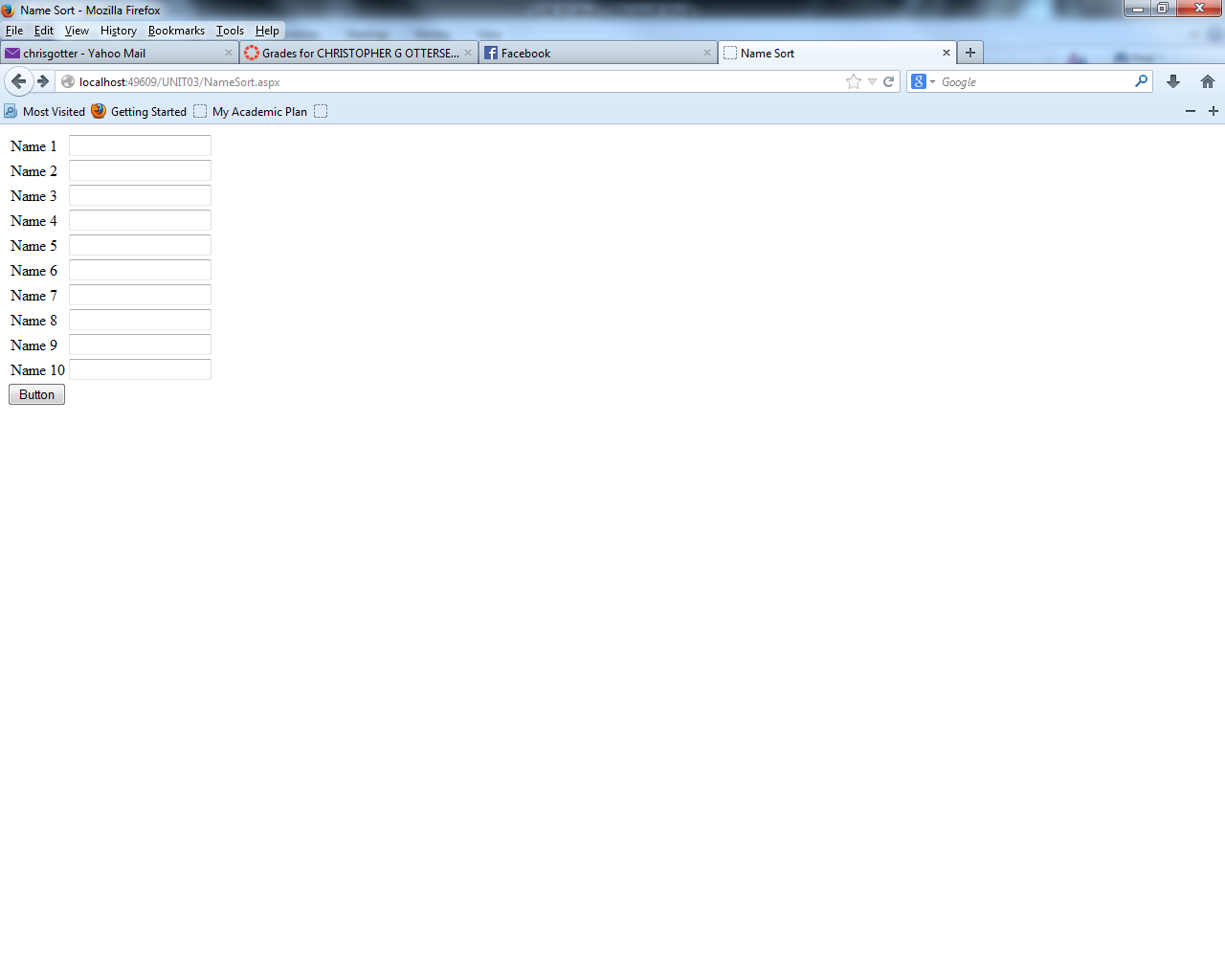
\*my bases are covered; unless I am mistaken I have about 6 pure html tables as well ☺

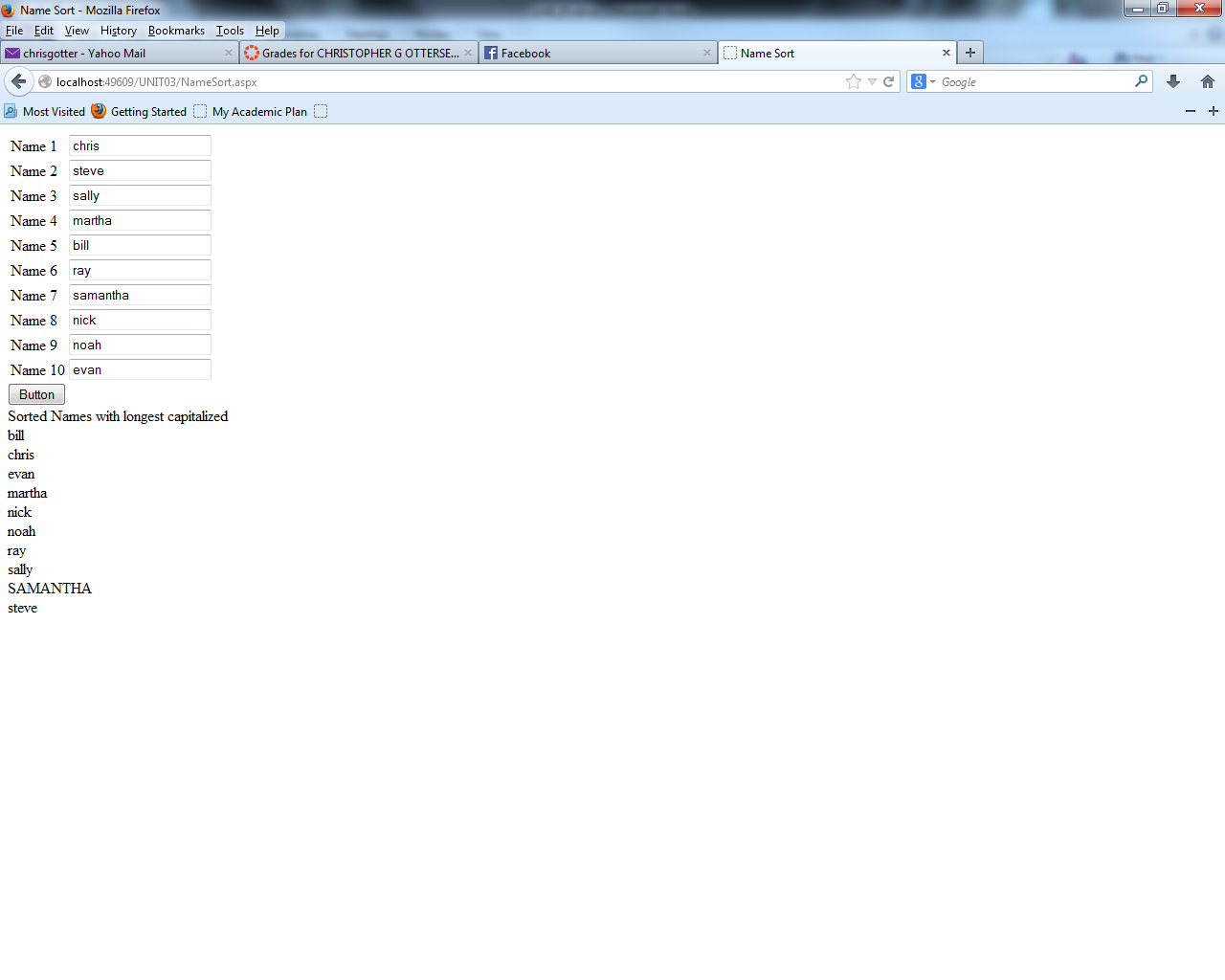
# PROGRAM OUTPUT

## Menu

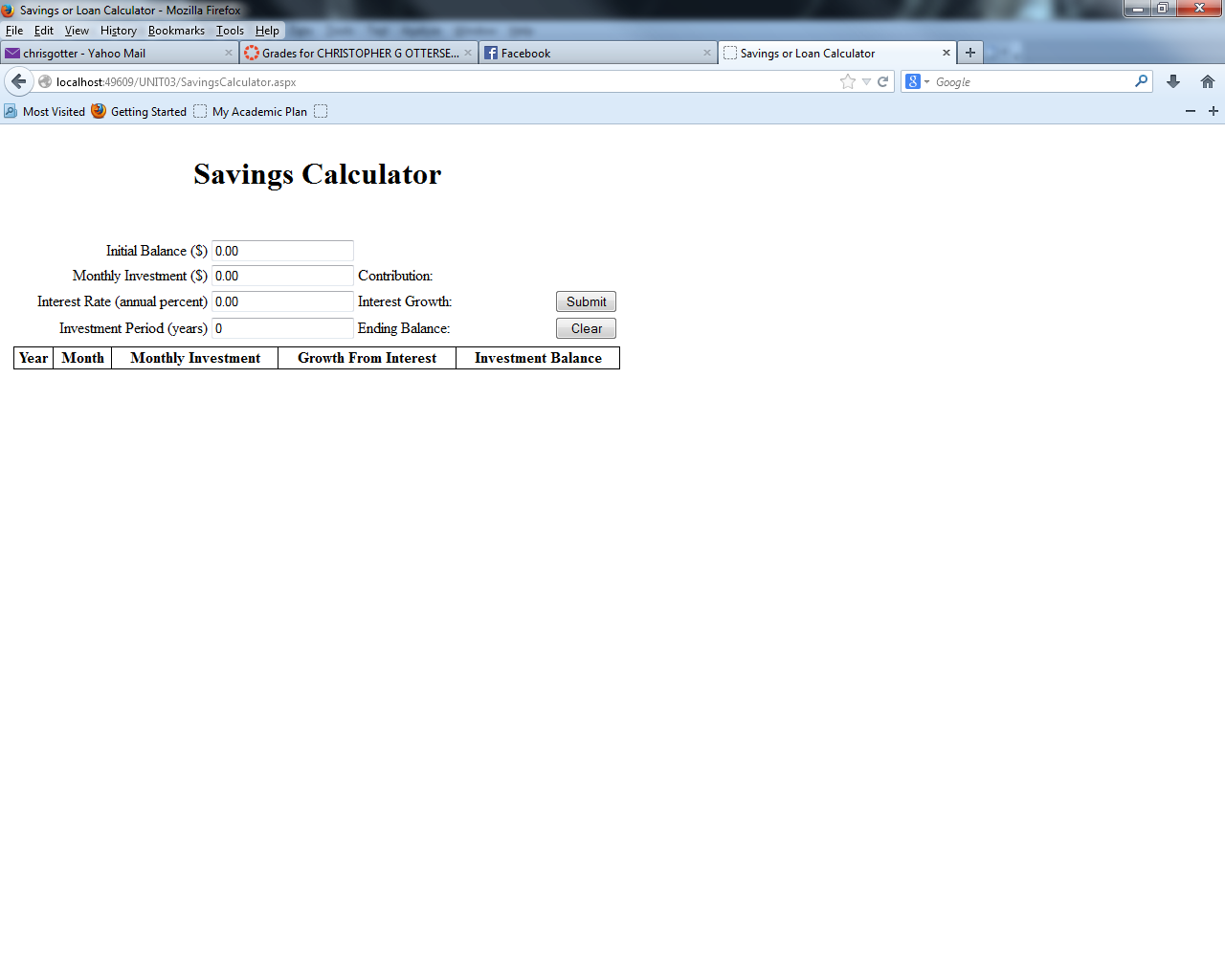


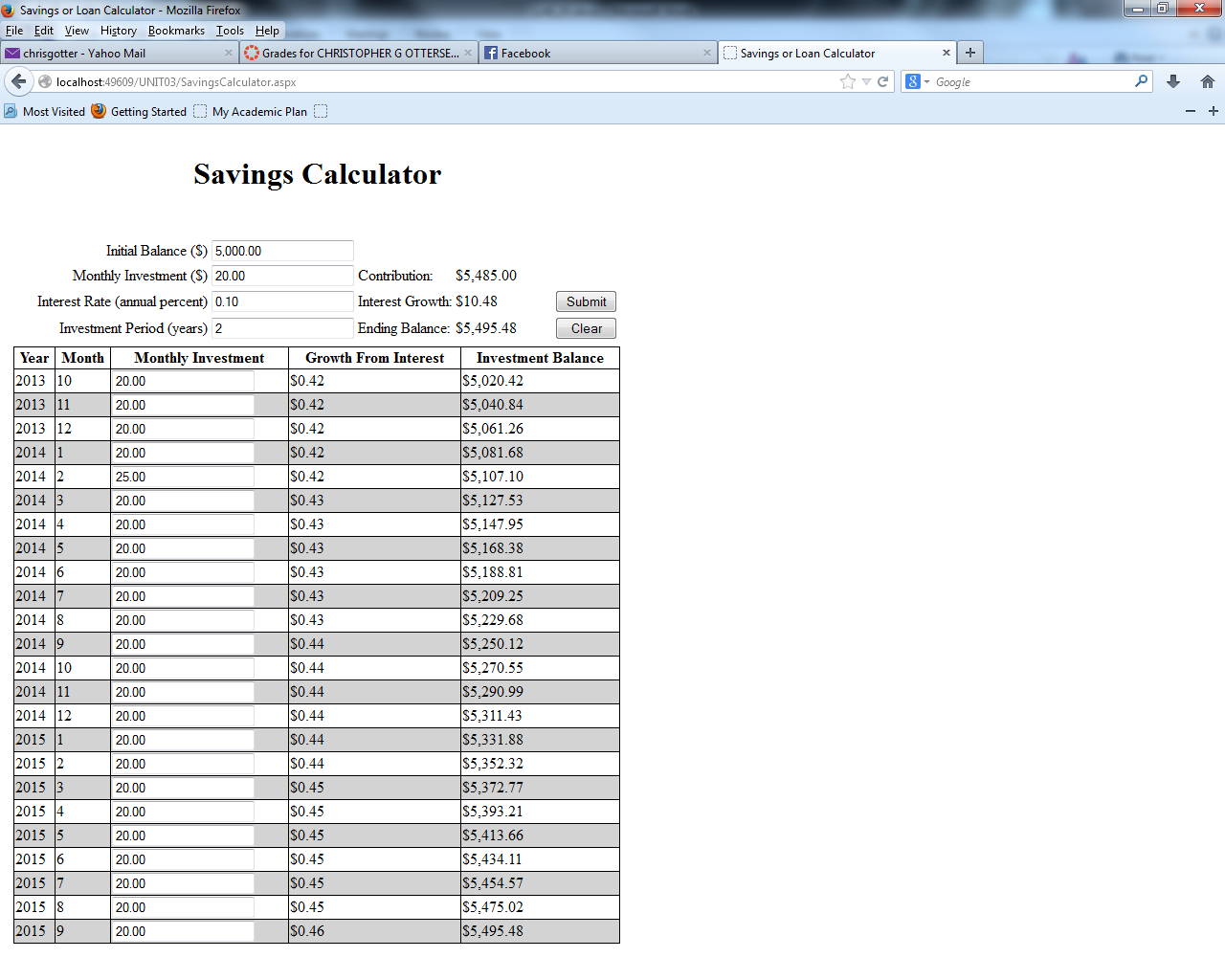
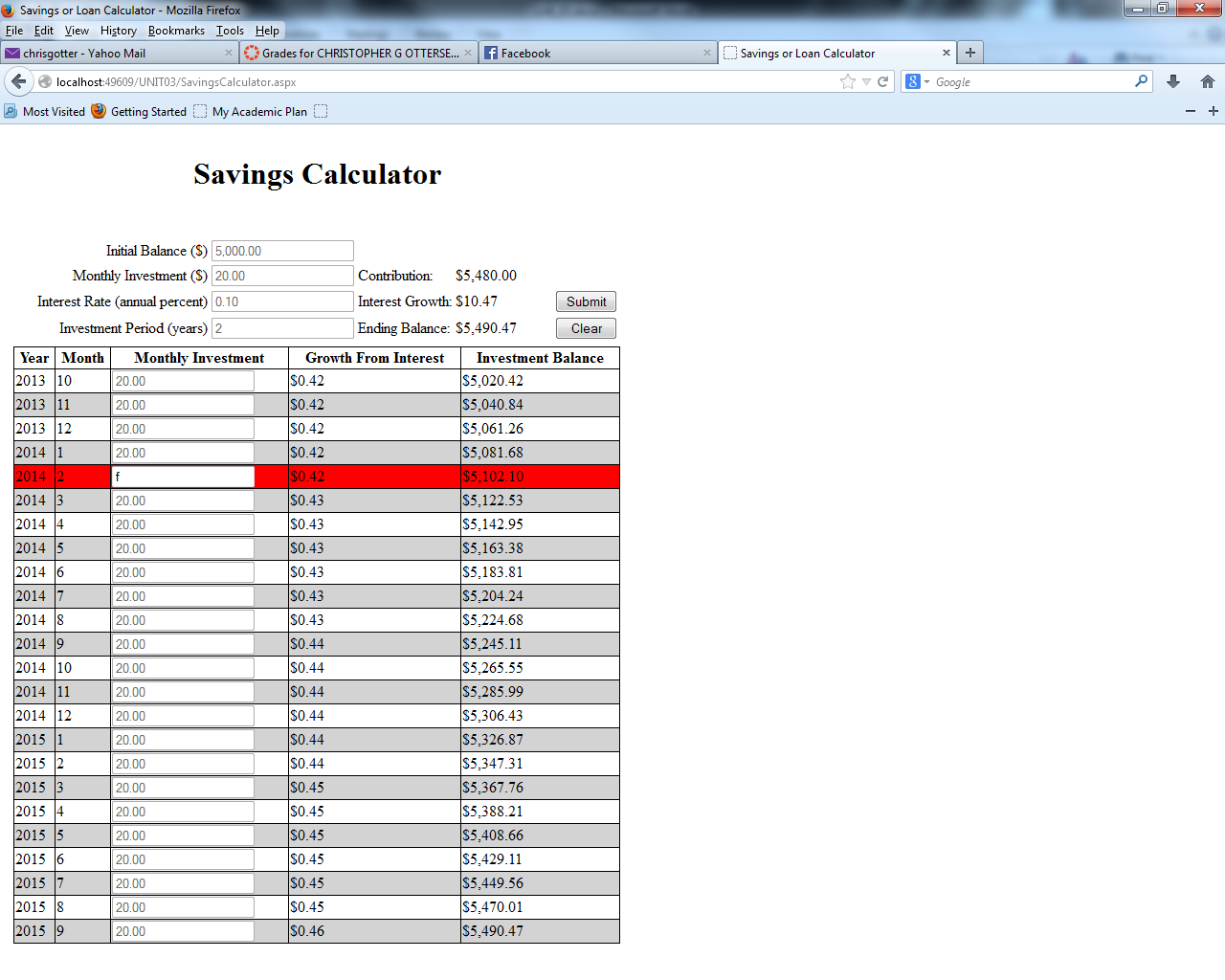
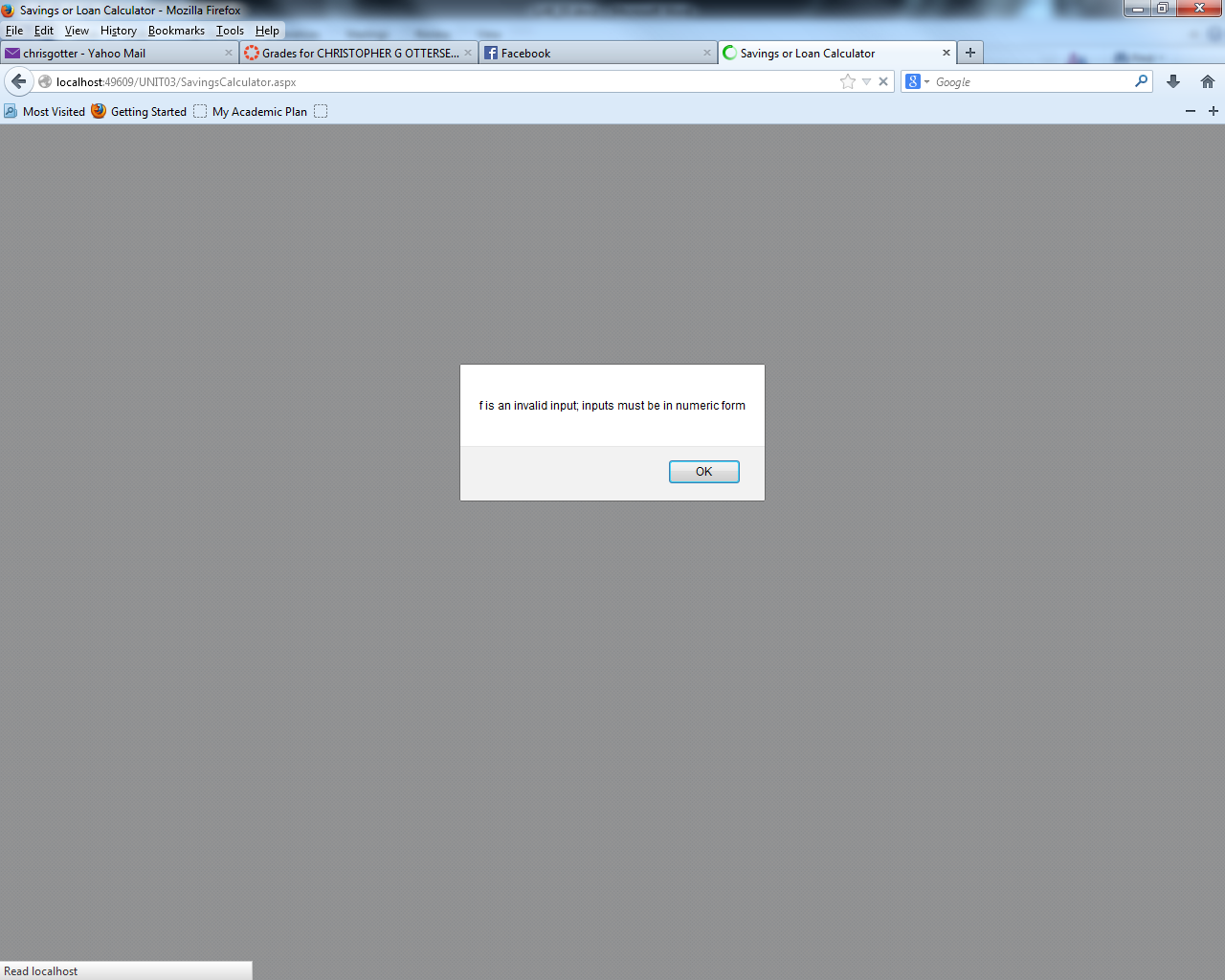
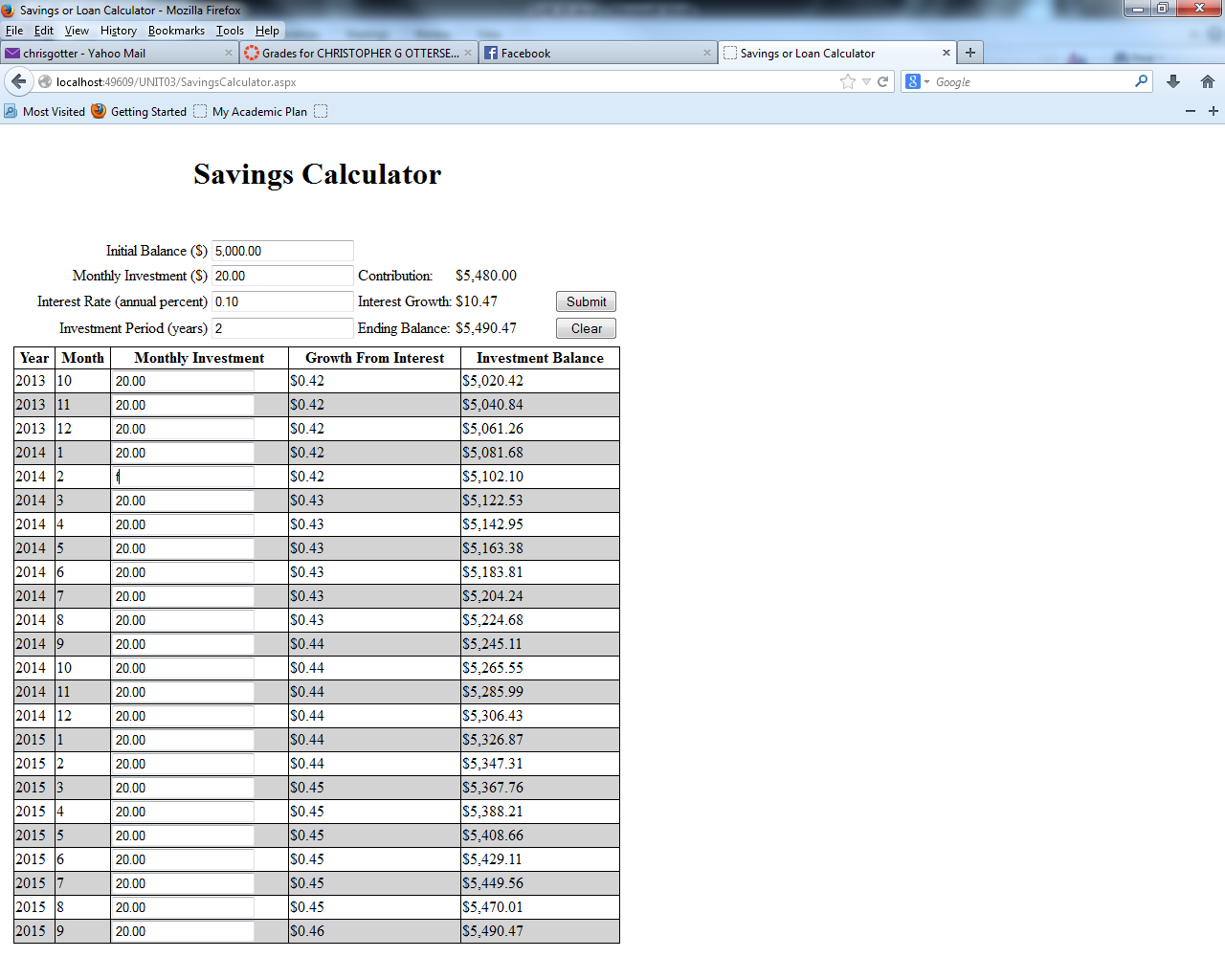
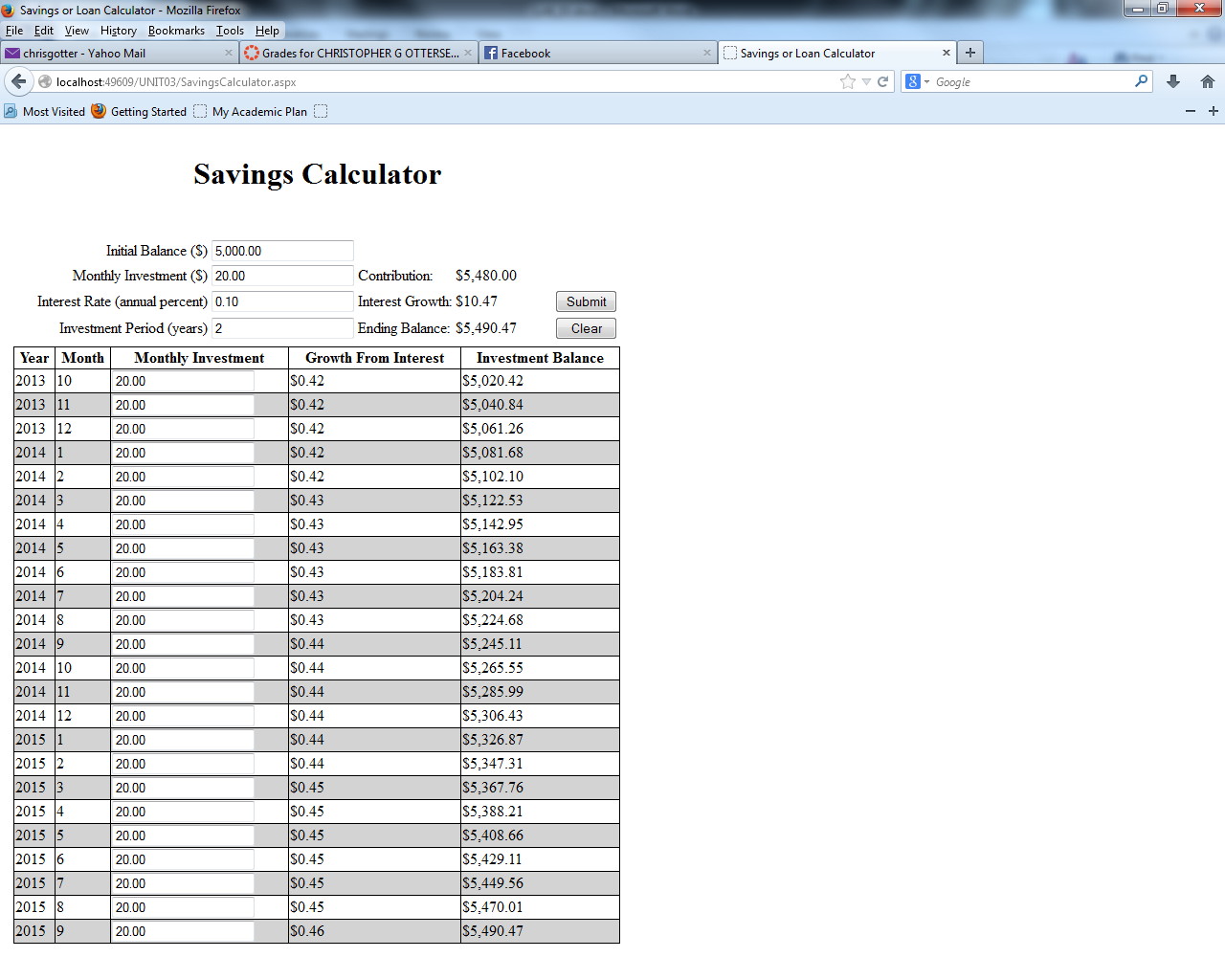
## NameSort





## Savings Calculator





# LINK TO RUN PROGRAM ON SERVER

<http://134.39.38.40/CIS219/cottersen/UNIT03/>

# SOURCE CODE

## Default.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="\_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head id="Head1" runat="server">

<title>Unit03</title>

</head>

<body>

<form id="form1" runat="server">

<h1>UNIT03 Homework</h1>

<div>

<asp:HyperLink ID="lnkNameSort" runat="server" NavigateUrl="~/NameSort.aspx" Text="Name Analysis">Name Analysis</asp:HyperLink>

<br />

<asp:HyperLink ID="lnkSavingsCalculator" runat="server" NavigateUrl="~/SavingsCalculator.aspx" Text="Savings Calculator">Credit Application</asp:HyperLink>

<br />

</div>

</form>

</body>

</html>

## Default.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

public partial class \_Default : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

}

## NameSort.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="NameSort.aspx.cs" Inherits="NameSort" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>Name Sort</title>

</head>

<body>

<form id="form1" runat="server">

<div>

<asp:Table ID="tbl\_input" runat="server">

</asp:Table>

<asp:Button ID="btn\_submit" runat="server" Text="Button"

onclick="btn\_submit\_Click" />

<asp:Label ID="lbl\_list" runat="server" Text=""></asp:Label>

</div>

</form>

</body>

</html>

## NameSort.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

public partial class NameSort : System.Web.UI.Page

{

private int inputCount = 1;

const int inputsDesired = 10;

List<Control[]> input\_array = new List<Control[]>(inputsDesired);

protected void Page\_Load(object sender, EventArgs e)

{

addBoxes(inputsDesired);

}

private void addBoxes(int argB)

{

while (inputCount <= argB)

{

Control[] control\_array =

{

new Label(),

new TextBox(),

new Label()

};

((Label)control\_array[0]).Text = "Name " + inputCount;

control\_array[0].ID = "lbl\_name" + inputCount;

control\_array[1].ID = "txt\_name" + inputCount;

control\_array[2].ID = "lbl\_err" + inputCount;

control\_array[2].Visible = false;

input\_array.Add(control\_array);

TableRow row = new TableRow();

foreach (Control control in control\_array)

{

TableCell cell = new TableCell();

cell.Controls.Add(control);

row.Cells.Add(cell);

}

tbl\_input.Rows.Add(row);

inputCount++;

}

}

protected void btn\_submit\_Click(object sender, EventArgs e)

{

List<string> names = new List<string>();

foreach (Control[] control in input\_array)

{

names.Add(((TextBox)control[1]).Text);

}

names.Sort();

largestToCaps(names);

displayNames(names);

}

private void displayNames(List<string> names)

{

lbl\_list.Text = "<br />" + "Sorted Names with longest capitalized";

foreach (string name in names)

{

lbl\_list.Text += (name != "" ? "<br />" : "") + name;

}

}

private static void largestToCaps(List<string> names)

{

List<int> largest = new List<int>(10);

largest.Add(0);

for (int i = 1; i < inputsDesired; i++)

{

if (names[i].Length > names[largest[0]].Length)

{

largest.Clear();

largest.Add(i);

}

else if (names[i].Length == names[largest[0]].Length)

{

largest.Add(i);

}

}

foreach (int i in largest)

{

names[i] = names[i].ToUpper();

}

}

}

## SavingsCalculator.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="SavingsCalculator.aspx.cs"

Inherits="SavingsCalculator" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>Savings or Loan Calculator</title>

<style type="text/css">

.ttl\_in

{

width: 200px;

text-align: right;

}

.ttl\_out

{

width: 100px;

}

</style>

</head>

<body>

<form id="form1" runat="server">

<div>

<table>

<tr>

<td align="center">

<h1>Savings Calculator</h1>

<br />

</td>

</tr>

<tr>

<td>

<table>

<tr>

<td>

<table style="width: 100%;">

<tr>

<td class="ttl\_in">

Initial Balance ($)

</td>

<td>

<asp:TextBox ID="txt\_init" runat="server" TextMode="Number" Text="0"></asp:TextBox>

</td>

<td >

</td>

<td class="ttl\_out">

</td>

<td >

</td>

</tr>

<tr>

<td class="ttl\_in">

Monthly Investment ($)

</td>

<td>

<asp:TextBox ID="txt\_inv" runat="server" TextMode="Number" Text="0"></asp:TextBox>

</td>

<td>

Contribution:

</td>

<td class="ttl\_out">

<asp:Label ID="lbl\_cont" runat="server"></asp:Label>

</td>

<td>

</td>

</tr>

<tr>

<td class="ttl\_in">

Interest Rate (annual percent)

</td>

<td>

<asp:TextBox ID="txt\_prc" runat="server" TextMode="Number" Text="0"></asp:TextBox>

</td>

<td>

Interest Growth:

</td>

<td class="ttl\_out">

<asp:Label ID="lbl\_intr" runat="server"></asp:Label>

</td>

<td>

<asp:Button ID="btn\_submit" runat="server" Text="Submit"

onclick="btn\_submit\_Click" />

</td>

</tr>

<tr>

<td class="ttl\_in">

Investment Period (years)

</td>

<td>

<asp:TextBox ID="txt\_per" runat="server" TextMode="Number" Text="0"></asp:TextBox>

</td>

<td>

Ending Balance:

</td>

<td class="ttl\_out">

<asp:Label ID="lbl\_tot" runat="server"></asp:Label>

</td>

<td>

<asp:Button ID="btn\_clear" runat="server" Text="Clear" Width="100%"

onclick="btn\_clear\_Click" />

</td>

</tr>

</table>

</td>

</tr>

<tr>

<td>

<asp:Table ID="tbl\_proj" runat="server" GridLines="Both" Width="100%">

<asp:TableHeaderRow>

<asp:TableHeaderCell Text="Year">

</asp:TableHeaderCell>

<asp:TableHeaderCell Text="Month">

</asp:TableHeaderCell>

<asp:TableHeaderCell Text="Monthly Investment">

</asp:TableHeaderCell>

<asp:TableHeaderCell Text="Growth From Interest">

</asp:TableHeaderCell>

<asp:TableHeaderCell Text="Investment Balance">

</asp:TableHeaderCell>

</asp:TableHeaderRow>

</asp:Table>

</td>

</tr>

</table>

</td>

</tr>

</table>

</div>

</form>

</body>

</html>

## SavingsCalculator.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

public partial class SavingsCalculator : System.Web.UI.Page

{

double monthlyInvestment;

int investmentPeriod;

double percentInterest;

private TextBox[] inputs = new TextBox[4];

bool busted = false;

//http://stackoverflow.com/questions/307737/asp-net-problem-with-event-handlers-for-dynamically-created-controls

EventHandler txt\_changed\_listener;

protected void txt\_TextChanged(object sender, EventArgs args)

{

monthlyInvestment = double.Parse(txt\_inv.Text);

double balance = double.Parse(txt\_init.Text);

for(int i = 1; i < tbl\_proj.Rows.Count; i++)

{

try

{

balance = populateRow(tbl\_proj.Rows[i], balance, i);

((TextBox)tbl\_proj.Rows[i].Cells[2].Controls[0]).Enabled = true;

}

catch (FormatException e)

{

lockdown(i);

return;

}

}

totals();

}

private void lockdown(int badVal)

{

foreach (TextBox input in inputs)

{

input.Enabled = false;

}

for (int j = 1; j < tbl\_proj.Rows.Count; j++)

{

((TextBox) tbl\_proj.Rows[j].Cells[2].Controls[0]).Enabled = false;

}

TextBox tb\_err;

tb\_err = ((TextBox)tbl\_proj.Rows[badVal].Cells[2].Controls[0]);

tbl\_proj.Rows[badVal].BackColor = System.Drawing.Color.Red;

tb\_err.Enabled = true;

tb\_err.Focus();

ClientScript.RegisterClientScriptBlock(this.GetType(), "alert",

"<script>alert('" + tb\_err.Text +

" is an invalid input; inputs must be in numeric form');</script>");

}

protected void Page\_Load(object sender, EventArgs args)

{

inputs[0] = txt\_init;

inputs[1] = txt\_inv;

inputs[2] = txt\_prc;

inputs[3] = txt\_per;

try

{

monthlyInvestment = double.Parse(txt\_inv.Text);

percentInterest = double.Parse(txt\_prc.Text)/12;

investmentPeriod = int.Parse(txt\_per.Text);

txt\_init.Text = double.Parse(txt\_init.Text).ToString("C").Substring(1);

txt\_inv.Text = double.Parse(txt\_inv.Text).ToString("C").Substring(1);

txt\_per.Text = int.Parse(txt\_per.Text).ToString();

txt\_prc.Text = double.Parse(txt\_prc.Text).ToString("C").Substring(1);

txt\_changed\_listener = new EventHandler(txt\_TextChanged);

populateTable();

busted = false;

}

catch (FormatException e)

{

busted = true;

}

}

private void totals()

{

double[] totals = {double.Parse(txt\_init.Text),0};

for (int i = 1; i < tbl\_proj.Rows.Count; i++)

{

TextBox tb = (TextBox)tbl\_proj.Rows[i].Cells[2].Controls[0];

tb.Text = double.Parse(tb.Text).ToString("C").Substring(1);

totals[0] += double.Parse(tb.Text);

totals[1] += double.Parse(tbl\_proj.Rows[i].Cells[3].ToolTip);

}

//lbl\_cont.Text = txt\_init.Text;

lbl\_cont.Text = totals[0].ToString("C");

lbl\_intr.Text = totals[1].ToString("C");

lbl\_tot.Text = double.Parse(tbl\_proj.Rows[tbl\_proj.Rows.Count-1].Cells[4].ToolTip).ToString("C");

}

protected void btn\_submit\_Click(object sender, EventArgs e)

{

btn\_clear\_Click(null, null);

if (!busted)

{

populateTable();

totals();

}

else

{

ClientScript.RegisterClientScriptBlock(this.GetType(), "alert",

"<script>alert('Invalid input; inputs must be in numeric form');</script>");

}

}

private void populateTable()

{

foreach (TextBox input in inputs)

{

input.Enabled = true;

}

double balance = double.Parse(txt\_init.Text);

DateTime period = DateTime.Now;

bool good\_input = true;

for (int i = 0; i < investmentPeriod \* 12; i++)

{

TableRow tr = new TableRow();

for (int j = 0; j < 5; j++)

{

tr.Cells.Add(new TableCell());

}

TextBox contribution = new TextBox();

contribution.ID = "row\_" + i;

contribution.Text = txt\_inv.Text;

contribution.AutoCompleteType = AutoCompleteType.None;

contribution.AutoPostBack = true;

contribution.TextChanged += txt\_changed\_listener;

contribution.Enabled = true;

tr.Cells[0].Text = period.Year.ToString();

tr.Cells[1].Text = period.Month.ToString();

tr.Cells[2].Controls.Add(contribution);

tbl\_proj.Rows.Add(tr);

tr.Cells[2].Enabled = true;

if (good\_input)

{

try

{

balance = populateRow(tr, balance, i+1);

}

catch (FormatException e)

{

good\_input = false;

}

}

period = period.AddMonths(1);

}

}

private double populateRow(TableRow row, double balance, int rowNum)

{

row.Cells[3].ToolTip = ((percentInterest / 100) \* balance).ToString();

row.Cells[3].Text = double.Parse(row.Cells[3].ToolTip).ToString("C");

balance = double.Parse(row.Cells[3].ToolTip) + balance + double.Parse(((TextBox)row.Cells[2].Controls[0]).Text);

row.Cells[4].Text = (balance).ToString("C");

row.Cells[4].ToolTip = (balance).ToString();

row.BackColor = rowNum%2==1?System.Drawing.Color.White:System.Drawing.Color.LightGray;

return balance;

}

protected void btn\_clear\_Click(object sender, EventArgs args)

{

while (tbl\_proj.Rows.Count > 1)

{

tbl\_proj.Rows.RemoveAt(1);

}

}

}

## web.config

<?xml version="1.0"?>

<!--

For more information on how to configure your ASP.NET application, please visit

http://go.microsoft.com/fwlink/?LinkId=169433

-->

<configuration>

<system.web>

<compilation debug="false" targetFramework="4.0" />

</system.web>

</configuration>