

**KAVIYATRI BAHINABAI CHAUDHARI,
NORTH MAHARASHTRA UNIVERSITY, JALGAON**

NES's

GANGAMAI COLLEGE OF ENGINEERING
ISO 9001:2008

COMPUTER ENGINEERING DEPARTMENT



Laboratory Manuals

Class: B.E. Computer (60-40 Pattern).

Semester: VIII

Subject: Advanced Technology II Laboratory.

Academic Year:2024-25.



**Nagaon Education Society's
GANGAMAI COLLEGE OF ENGINEERING,
NAGAON, DHULE-05
Computer Department**

Institute Vision

- Empowering first generation engineers to excel in technical education based on human values

Institute Mission

- To impart affordable and quality education in order to meet needs of industry and to achieve excellence in teaching learning process.
- To achieve excellence in application oriented research in selected area of Technology to contribute to the development of the region.
- To collaborate with industries to promote innovation capabilities of budding engineers.
- To develop responsible citizens to awareness and acceptance of ethical values.
- To build a support system of all stakeholders to develop the institute.



**Nagaon Education Society's
GANGAMAI COLLEGE OF ENGINEERING,
NAGAON, DHULE-05
Computer Department**

DEPARTMENT OF COMPUTER ENGINEERING

Vision

To emerge as the leading Computer Engineering department for inclusive development of students.

Mission

To provide student-centered conducive environment for preparing knowledgeable, competent and value-added computer engineers.

A Laboratory Manual

For

Advanced Technology II – Lab

For the Bachelor of Engineering in the Computer Engineering

BE Computer

Semester – VIII

2024-2025

Name:

Roll No:.....Batch:.....

PRN NO:.....



Nagaon Education Society's
GANGAMAI COLLEGE OF ENGINEERING,
NAGAON, DHULE-05
Computer Department

CERTIFICATE

This is to certify that

Mr./Mrs. _____

Having Roll No. ____

Of VIII Semester for the course Bachelor of Computer Engineering
Of the institute **Gangamai College of Engineering, Nagaon, Dhule**, has
completed the term work satisfactorily of the subject

Engineering - Advanced Technology II Lab

for the academic year 2024 – 2025

as prescribed in the curriculum

Date: _____

PRN No: _____

Place: Nagaon, Dhule

Exam Seat No: _____

Subject Teacher

HOD

Principal

Seal of the Institute

PRACTICAL-COURSE OUTCOMES

COURSE OUTCOMES(CO_s)

1. Break down real world problems / application.
2. Demonstrate Full Stack development.
3. Design Full Stack based applications.
4. Decide tools for Full Stack development.
5. Develop Full Stack based applications.

Expt No.	Name of Experiment	Page No.	Starting Date	Ending Date	Remark
1.	Data Visualization using Python.				
2.	Implementation of ASP.net Application with Sql Server as back-end .				
3.	Create a Ruby on Rails an application (Use Technology)				

DEPARTMENT OF COMPUTER ENGINEERING

Objectives

To enhance competency by undertaking laboratory assignments using Full Stack.

DEPARTMENT OF COMPUTER ENGINEERING

Programme Educational Objectives

PEO 1.Core Knowledge

Computer engineering graduates will have the knowledge of basic science and Engineering skills, Humanities, social science, management and conceptual and practical understanding of core computer engineering area with project development.

PEO 2.Employment

Computer engineering graduates will have the knowledge of Industry-based technical skills to succeed in entry level engineering position at various industries as well as in academics.

PEO 3. Professional Competency

Computer engineering graduates will have the ability to communicate effectively in English, to accumulate and disseminate the knowledge and to work effectively in a team with a sense of social awareness.



**GANGAMAI COLLEGE OF ENGINEERING, NAGAON.
DEPARTMENT OF COMPUTER**

Name: _____

Class: B.E. Computer

Division :

Batch:

Roll No:

Subject: **Advance Technology Lab II**

Date of Performance: __/__/20__

Date of Completion: __/__/20__

Grade:

Sign:

Experiment No. 1

Aim: Data Visualization using Python.

1. Objective: To demonstrate various python libraries such as Matplotlib , Seaborn, Pandas

2. Background:

Data visualization is the presentation of data in a pictorial or graphical format. It enables decision makers to see [analytics](#) presented visually, so they can grasp difficult concepts or identify new patterns. With interactive visualization, you can take the concept a step further by using technology to drill down into charts and graphs for more detail, interactively changing what data you see and how it's processed.

History of Data Visualization

The concept of using pictures to understand data has been around for centuries, from maps and graphs in the 17th century to the invention of the pie chart in the early 1800s. Several decades later, one of the most cited examples of statistical graphics occurred when Charles Minard mapped Napoleon's invasion of Russia. The map depicted the size of the army as well as the path of Napoleon's retreat from Moscow – and tied that information to temperature and time scales for a more in-depth understanding of the event. It's technology, however, that truly lit the fire under data visualization. Computers made it possible to process large amounts of data at lightning-fast speeds. Today, data visualization has become a rapidly evolving blend of science and art that is certain to change the corporate landscape over the next few years.

Why is data visualization important?

Because of the way the human brain processes information, using charts or graphs to visualize large amounts of complex data is easier than poring over spreadsheets or reports. Data visualization is a quick, easy way to convey concepts in a universal manner – and you can experiment with different scenarios by making slight adjustments.

Data visualization can also:

- Identify areas that need attention or improvement.
- Clarify which factors influence customer behavior.
- Help you understand which products to place where.
- Predict sales volumes.

Python offers multiple great graphing libraries that come packed with lots of different features. No matter if you want to create interactive, live or highly customized plots python has an excellent library for you.

To get a little overview here are a few popular plotting libraries: Matplotlib, Pandas and Seaborn

Outcomes:

Able to understand :

- Matplotlib for low level, provides lots of freedom to visualization
- Pandas Visualization: easy to use interface, built on Matplotlib
- Seaborn: high-level interface, great default styles

Example:

Data Visualization dataset: San Francisco Salaries

```
salaries = pd.read_csv('./Salaries.csv')

salaries.info()

for col in ['BasePay', 'OvertimePay', 'OtherPay', 'Benefits']:

    salaries[col]=pd.to_numeric(salaries[col],errors='coerce')

pay_columns=salaries.columns[3:salaries.columns.get_loc('Year')]

pay_columns

pays_arrangement = list(zip(*(iter(pay_columns),) * 3)) fig,
```

```

axes = plt.subplots(2,3)

for i in range(len(pays_arrangement)):
    for j in range(len(pays_arrangement[i])):
        # pass in axes to pandas hist
        salaries[pays_arrangement[i][j]].hist(ax=axes[i,j]) #
        axis objects have a lot of methods for customizing the look of a plot
        axes[i,j].set_title(pays_arrangement[i][j])plt.show()

fig, axes = plt.subplots(2,3) #
set the figure height
fig.set_figheight(5)fig.set_figwidth(12) for
i in range(len(pays_arrangement)):
    for j in range(len(pays_arrangement[i])):#pass in axes to pandas hist
        salaries[pays_arrangement[i][j]].hist(ax=axes[i,j]) axes[i,j].set_title(pays_arrangement[i][j])

# add a row of emptiness between the two rows
plt.subplots_adjust(hspace=1)

# add a row of emptiness between the cols
plt.subplots_adjust(wspace=1)

plt.show()

# and here is a cleaner version using tick rotation and plot spacing fig,
axes = plt.subplots(2,3)

# set the figure heightfig.set_figheight(5)
fig.set_figwidth(12)

for i in range(len(pays_arrangement)):

    for j in range(len(pays_arrangement[i])):

        salaries[pays_arrangement[i][j]].hist(ax=axes[i,j])

```

```
axes[i,j].set_title(pays_arrangement[i][j])

# set xticks with these labels, axes[i,j].set_xticklabels(labels=axes[i,j].get_xticks(),

#withthisrotation

rotation=30)

plt.subplots_adjust(hspace=1)

plt.subplots_adjust(wspace=1)

plt.show()
```

Questions and Answers:

Questions: 1. What type of Variables uses in Python?

Questions: 1. Explain Datatype uses in Python?

Questions: 1. How Database Connectivity done in Python?

Handwritten Answer on Next Blank Page of same margin



**GANGAMAI COLLEGE OF ENGINEERING, NAGAON.
DEPARTMENT OF COMPUTER**

Name: _____

Class: B.E. Computer

Division :

Batch:

Roll No:

Subject: **Advance Technology Lab II**

Date of Performance: __/__/20__

Date of Completion: __/__/20__

Grade:

Sign:

Experiment No. 2

Aim: Implementation of ASP.net Application with Sql Server as back-end .

1. Objective: Development and deployment of ASP.net Application software.

2. Background:

ASP.NET is an open-source,^[2] server-side web-application framework designed for web development to produce dynamic web pages. It was developed by Microsoft to allow programmers to build dynamic web sites, applications and services.

It was first released in January 2002 with version 1.0 of the .NET Framework and is the successor to Microsoft's Active Server Pages (ASP) technology. ASP.NET is built on the Common Language Runtime (CLR), allowing programmers to write ASP.NET code using any supported .NET language. The ASP.NET SOAP extension framework allows ASP.NET components to process SOAP messages.

ASP.NET's successor is ASP.NET Core. It is a re-implementation of ASP.NET as a modular web framework, together with other frameworks like Entity Framework. The new framework uses the new open-source .NET Compiler Platform (codename "Roslyn") and is cross platform. ASP.NET MVC, ASP.NET Web API, and ASP.NET Web Pages (a platform using only Razor pages) have merged into a unified MVC

ASP.net and SqlServer (ADO.net)

ADO.NET provides a bridge between the front end controls and the back end database. The ADO.NET objects encapsulate all the data access operations and the controls interact with these objects to display data, thus hiding the details of movement of data.

The following figure shows the ADO.NET objects at a glance:

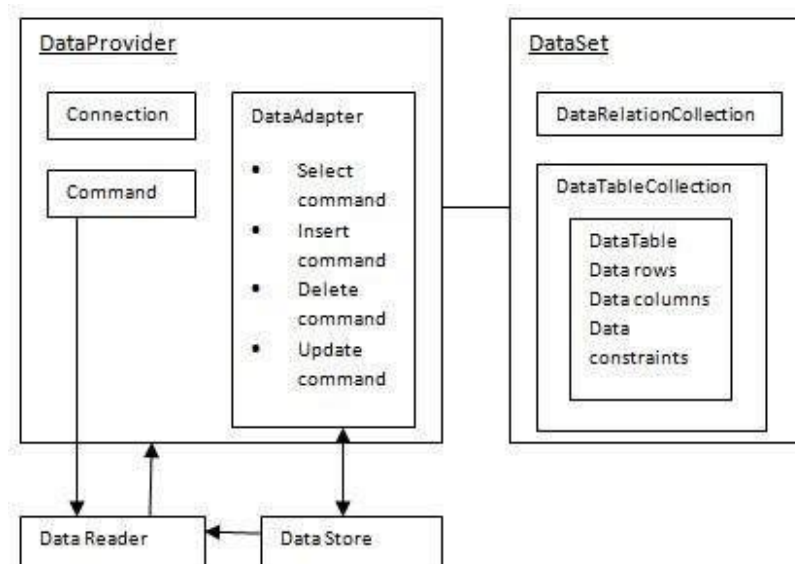


Fig :ADO.NET Objects

The DataSet Class

The dataset represents a subset of the database. It does not have a continuous connection to the database. To update the database a reconnection is required. The DataSet contains DataTable objects and DataRelation objects. The DataRelation objects represent the relationship between two tables.

Following table shows some important properties of the DataSet class:

Properties	Description
CaseSensitive	Indicates whether string comparisons within the data tables are case-sensitive.
Container	Gets the container for the component.

DataSetName	Gets or sets the name of the current data set.
DefaultViewManager	Returns a view of data in the data set.
DesignMode	Indicates whether the component is currently in design mode.
EnforceConstraints	Indicates whether constraint rules are followed when attempting any update operation.
Events	Gets the list of event handlers that are attached to this component.
ExtendedProperties	Gets the collection of customized user information associated with the DataSet.
HasErrors	Indicates if there are any errors.
IsInitialized	Indicates whether the DataSet is initialized.
Locale	Gets or sets the locale information used to compare strings within the table.
Namespace	Gets or sets the namespace of the DataSet.
Prefix	Gets or sets an XML prefix that aliases the namespace of the DataSet.
Relations	Returns the collection of DataRelation objects.
Tables	Returns the collection of DataTable objects.

The following table shows some important methods of the DataSet class:

Methods	Description
AcceptChanges	Accepts all changes made since the DataSet was loaded or this method was called.
BeginInit	Begins the initialization of the DataSet. The initialization occurs at run time.
Clear	Clears data.
Clone	Copies the structure of the DataSet, including all DataTable schemas, relations, and constraints. Does not copy any data.
Copy	Copies both structure and data.
CreateDataReader()	Returns a DataTableReader with one result set per DataTable, in the same sequence as the tables appear in the Tables collection.
CreateDataReader(DataTable[])	Returns a DataTableReader with one result set per DataTable.
EndInit	Ends the initialization of the data set.
Equals(Object)	Determines whether the specified Object is equal to the current Object.
Finalize	Free resources and perform other cleanups.
GetChanges	Returns a copy of the DataSet with all changes

	made since it was loaded or the AcceptChanges method was called.
GetChanges(DataRowState)	Gets a copy of DataSet with all changes made since it was loaded or the AcceptChanges method was called, filtered by DataRowState.
GetDataSetSchema	Gets a copy of XmlSchemaSet for the DataSet.
GetObjectData	Populates a serialization information object with the data needed to serialize the DataSet.
GetType	Gets the type of the current instance.
GetXML	Returns the XML representation of the data.
GetXMLSchema	Returns the XSD schema for the XML representation of the data.
HasChanges()	Gets a value indicating whether the DataSet has changes, including new, deleted, or modified rows.
HasChanges(DataRowState)	Gets a value indicating whether the DataSet has changes, including new, deleted, or modified rows, filtered by DataRowState.
IsBinarySerialized	Inspects the format of the serialized representation of the DataSet.

Load(IDataReader, LoadOption, DataTable[])	Fills a DataSet with values from a data source using the supplied IDataReader, using an array of DataTable instances to supply the schema and namespace information.
Load(IDataReader, LoadOption, String[])	Fills a DataSet with values from a data source using the supplied IDataReader, using an array of strings to supply the names for the tables within the DataSet.
Merge()	Merges the data with data from another DataSet. This method has different overloaded forms.
ReadXML()	Reads an XML schema and data into the DataSet. This method has different overloaded forms.
ReadXMLSchema()	Reads an XML schema into the DataSet. This method has different overloaded forms.
RejectChanges	Rolls back all changes made since the last call to AcceptChanges.
WriteXML()	Writes an XML schema and data from the DataSet. This method has different overloaded forms.
WriteXMLSchema()	Writes the structure of the DataSet as an XML schema. This method has different overloaded forms.

The DataTable Class

The DataTable class represents the tables in the database. It has the following important properties; most of these properties are read only properties except the PrimaryKey property:

Properties	Description
ChildRelations	Returns the collection of child relationship.
Columns	Returns the Columns collection.
Constraints	Returns the Constraints collection.
DataSet	Returns the parent DataSet.
DefaultView	Returns a view of the table.
ParentRelations	Returns the ParentRelations collection.
PrimaryKey	Gets or sets an array of columns as the primary key for the table.
Rows	Returns the Rows collection.

The following table shows some important methods of the DataTable class:

Methods	Description
AcceptChanges	Commits all changes since the last AcceptChanges.
Clear	Clears all data from the table.

GetChanges	Returns a copy of the DataTable with all changes made since the AcceptChanges method was called.
GetErrors	Returns an array of rows with errors.
ImportRows	Copies a new row into the table.
LoadDataRow	Finds and updates a specific row, or creates a new one, if not found any.
Merge	Merges the table with another DataTable.
NewRow	Creates a new DataRow.
RejectChanges	Rolls back all changes made since the last call to AcceptChanges.
Reset	Resets the table to its original state.
Select	Returns an array of DataRow objects.

The DataRow Class

The DataRow object represents a row in a table. It has the following important properties:

Properties	Description
HasErrors	Indicates if there are any errors.
Items	Gets or sets the data stored in a specific column.
ItemArrays	Gets or sets all the values for the row.

Table	Returns the parent table.
-------	---------------------------

The following table shows some important methods of the DataRow class:

Methods	Description
AcceptChanges	Accepts all changes made since this method was called.
BeginEdit	Begins edit operation.
CancelEdit	Cancels edit operation.
Delete	Deletes the DataRow.
EndEdit	Ends the edit operation.
GetChildRows	Gets the child rows of this row.
GetParentRow	Gets the parent row.
GetParentRows	Gets parent rows of DataRow object.
RejectChanges	Rolls back all changes made since the last call to AcceptChanges.

The DataAdapter Object

The DataAdapter object acts as a mediator between the DataSet object and the database. This helps the Dataset to contain data from multiple databases or other data source.

The DataReader Object

The DataReader object is an alternative to the DataSet and DataAdapter combination. This object provides a connection oriented access to the data records in the database. These objects are suitable for read-only access, such as populating a list and then breaking the connection.

DbCommand and DbConnection Objects

The DbConnection object represents a connection to the data source. The connection could be shared among different command objects.

The DbCommand object represents the command or a stored procedure sent to the database from retrieving or manipulating data.

3. Pre-lab Task:

1. Use case diagram for College management system
(Here Draw Diagram)
2. Class Diagram for College management system
(Here Draw Diagram)
3. E-R diagram for College management system
(Here Draw Diagram)

4. Post Lab Task

Login Form

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true"
CodeBehind="Login.aspx.cs" Inherits="WebApplication2.WebForm2" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
    <style type="text/css">
        .style1
        {
            width: 74%;
            border-collapse: collapse;
            border: 1px solid #808000;
            margin-left: 0px;
            background-color: #C0C0C0;
        }
        .style2
        {
            width: 174px;
            color: #800000;
            font-family: "Times New Roman", Times, serif;
            font-size: x-large;
        }
        .style3
        {
            width: 7px;
            text-align: center;
        }
        .style4
        {
            width: 375px;
        }
    </style>
</asp:Content>
```

```

        .style5
        {
            width: 174px;
            color: #CC0000;
            font-size: small;
        }
        .style6
        {
            width: 174px;
            color: #FF0000;
            height: 28px;
        }
        .style7
        {
            width: 7px;
            text-align: center;
            height: 28px;
        }
        .style8
        {
            width: 375px;
            height: 28px;
        }
        .style10
        {
            width: 174px;
            color: #CC0000;
            font-size: large;
            text-align: right;
        }
        .style11
        {
            width: 174px;
            color: #FF0000;
            font-family: "Times New Roman", Times, serif;
            font-size: x-large;
            text-align: right;
        }
    </style>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="MainContent" runat="server">
    <table align="center" class="style1">
        <tr>
            <td class="style5">
                &nbsp;</td>
            <td class="style3">
                &nbsp;</td>
            <td class="style4">
                <asp:Label ID="lbl" runat="server"
                    style="color: #FF0000; font-size: x-large; font-weight: 700"
Text="Label"></asp:Label>
                </td>
            </tr>
            <tr>
                <td class="style10">
                    <strong>User Name</strong></td>
                <td class="style3">
                    :</td>

```

```

        <td class="style4">
            <asp:TextBox ID="txtUserName" runat="server"></asp:TextBox>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
                ErrorMessage="Enter User Name"
ControlToValidate="txtUserName"></asp:RequiredFieldValidator>
        </td>
    </tr>
    <tr>
        <td class="style11">
            <strong>Password</strong></td>
        <td class="style3">
            :</td>
        <td class="style4">
            <asp:TextBox ID="txtPassword" runat="server"
TextMode="Password"></asp:TextBox>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server"
                ErrorMessage="Enter Passwrđ"
ControlToValidate="txtUserName"></asp:RequiredFieldValidator>
        </td>
    </tr>
    <tr>
        <td class="style6">
        </td>
        <td class="style7">
        </td>
        <td class="style8">
            <asp:Button ID="btnSubmit" runat="server" Text="Submit"
                onclick="btnSubmit_Click" />
            <asp:ValidationSummary ID="ValidationSummary1" runat="server"
                ShowMessageBox="True" />
        </td>
    </tr>
</table>
</asp:Content>

```

Login Form Asp.net Code:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;

namespace WebApplication2
{
    public partial class WebForm2 : System.Web.UI.Page
    {
        SqlConnection con = new SqlConnection("Data Source=.\SQLExpress;Initial
Catalog=sinup;User ID=sa;Password=ssbt");
        protected void Page_Load(object sender, EventArgs e)
        {
            lbl.Visible = false;
        }
        public void show()
    }
}

```



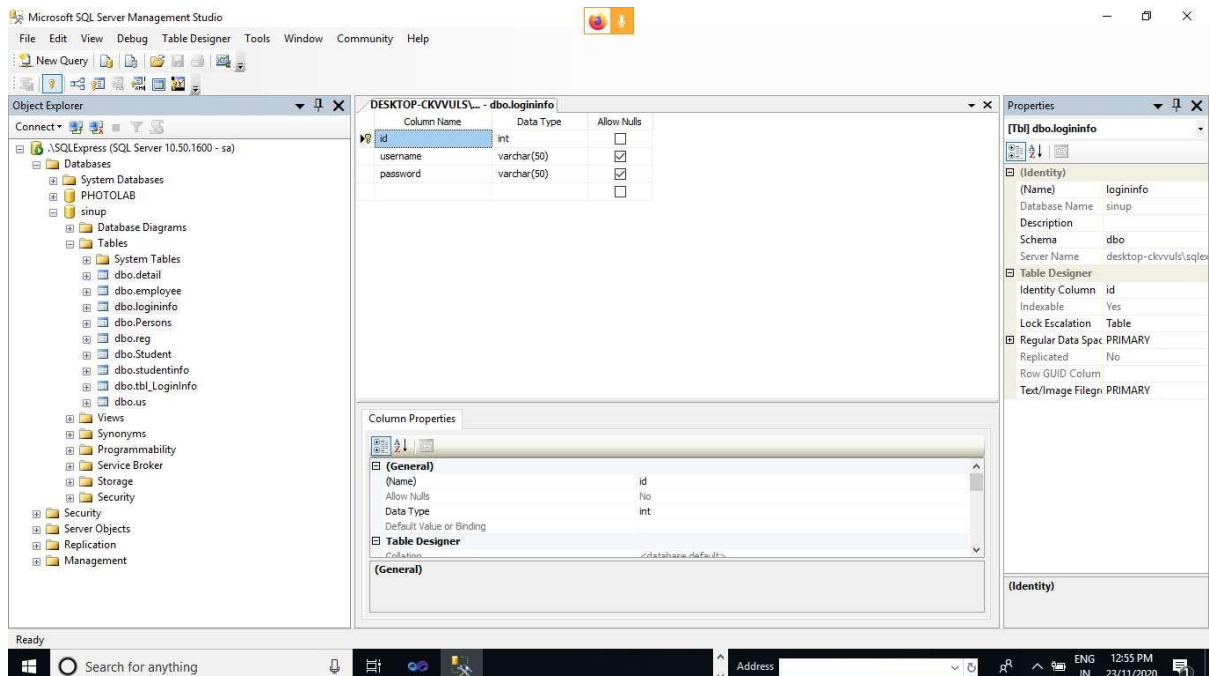
```

    {
        try
        {
            SqlCommand sqlcmd = new SqlCommand("insert into logininfo values('" +
txtUserName.Text + "',''" + txtPassword.Text + "')", con);
            con.Open();
            sqlcmd.ExecuteNonQuery();
            con.Close();
        }
        catch (Exception ex)
        {
            lbl.Text = ex.Message;
        }
        finally
        {
            con.Close();
            con.Dispose();
        }
    }

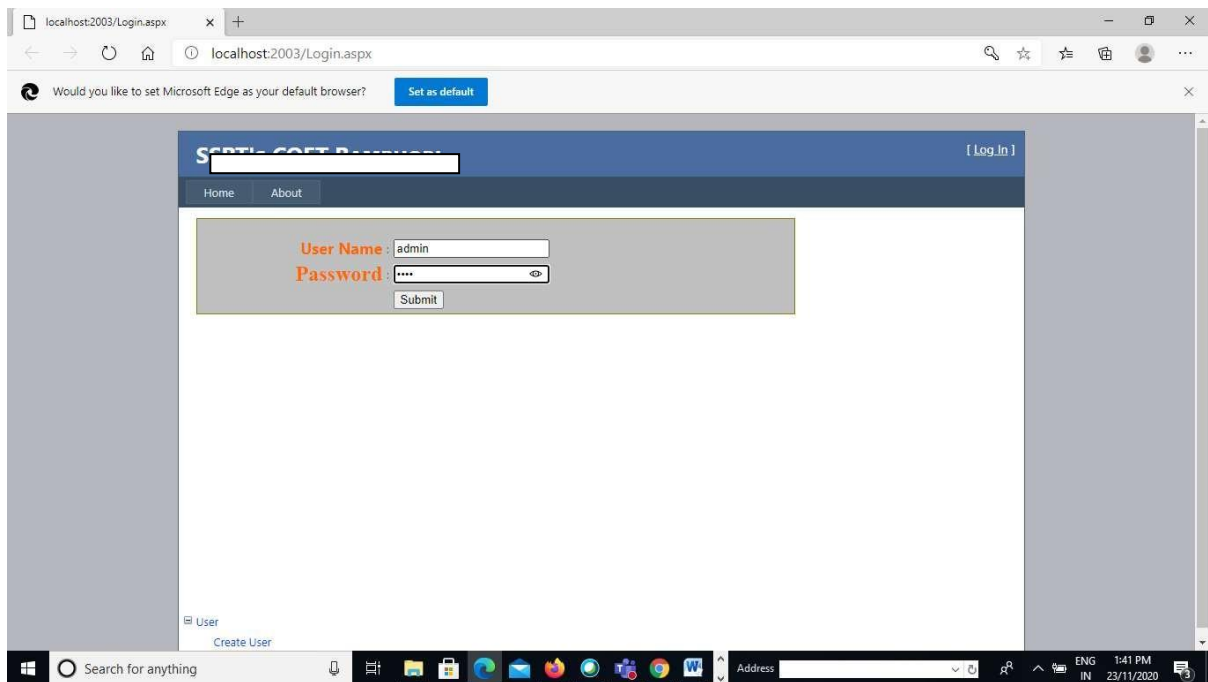
protected void btnSubmit_Click(object sender, EventArgs e)
{
    show();
}
}
}

```

Database



Output



Information Form Design

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true"
CodeBehind="info.aspx.cs" Inherits="WebApplication2.WebForm4" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
    <style type="text/css">
        .style1
        {
            width: 100%;
        }
        .style2
        {
            width: 41px;
            text-align: center;
        }
        .style3
        {
            width: 232px;
        }
        .style4
        {
            font-family: "Times New Roman";
            font-weight: bold;
        }
    </style>
</asp:Content>
```

```

        color: #00FF00;
        font-size: x-large;
        background-color: #CC0000;
    }
    .style5
    {
        background-color: #CC3300;
    }
    .style6
    {
        width: 41px;
        text-align: center;
        color: #000000;
        font-size: x-large;
        background-color: #669900;
    }
</style>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="MainContent" runat="server">
    <table class="style1">
        <tr>
            <td class="style4" colspan="3" width="30">
                <asp:Label ID="lbl" runat="server"></asp:Label>
            </td>
        </tr>
        <tr>
            <td class="style4" width="30">
                Roll No</td>
            <td class="style6">
                :</td>
            <td class="style5">
                <asp:TextBox ID="txtRoll" runat="server"></asp:TextBox>
            </td>
        </tr>
        <tr>
            <td class="style4" width="30">
                Student Name</td>
            <td class="style6">
                :</td>
            <td class="style5">
                <asp:TextBox ID="txtStudentName" runat="server"></asp:TextBox>
            </td>
        </tr>
        <tr>
            <td class="style4" width="30">
                Father Name</td>
            <td class="style6">
                :</td>
            <td class="style5">
                <asp:TextBox ID="txtFatherName" runat="server"></asp:TextBox>
            </td>
        </tr>
        <tr>
            <td class="style4" width="30">
                City</td>
            <td class="style6">
                :</td>
            <td class="style5">

```



```

sqlcom.ExecuteNonQuery();
con.Close();
lbl.Text = "Record Inserted Successfully";
lbl.Visible = true;
clearcontrol();

}

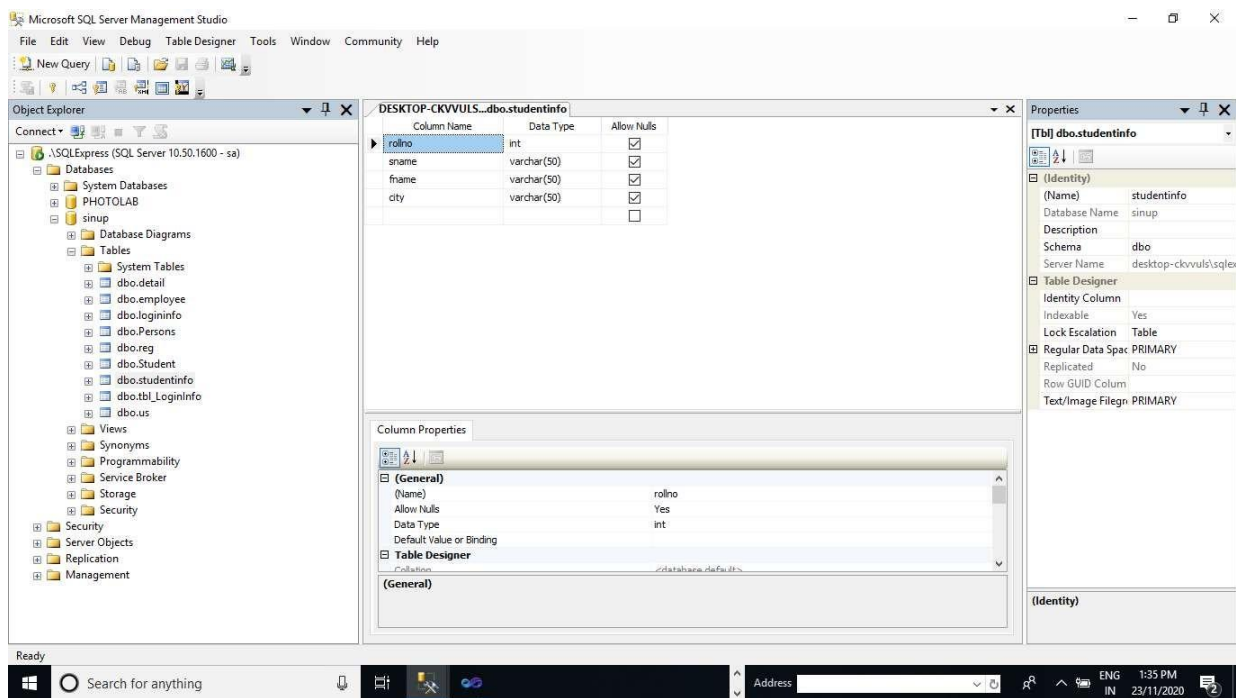
public void clearcontrol()
{
    txtCity.Text = "";
    txtFatherName.Text = "";
    txtRoll.Text = "";
    txtStudentName.Text = "";
}

}
}

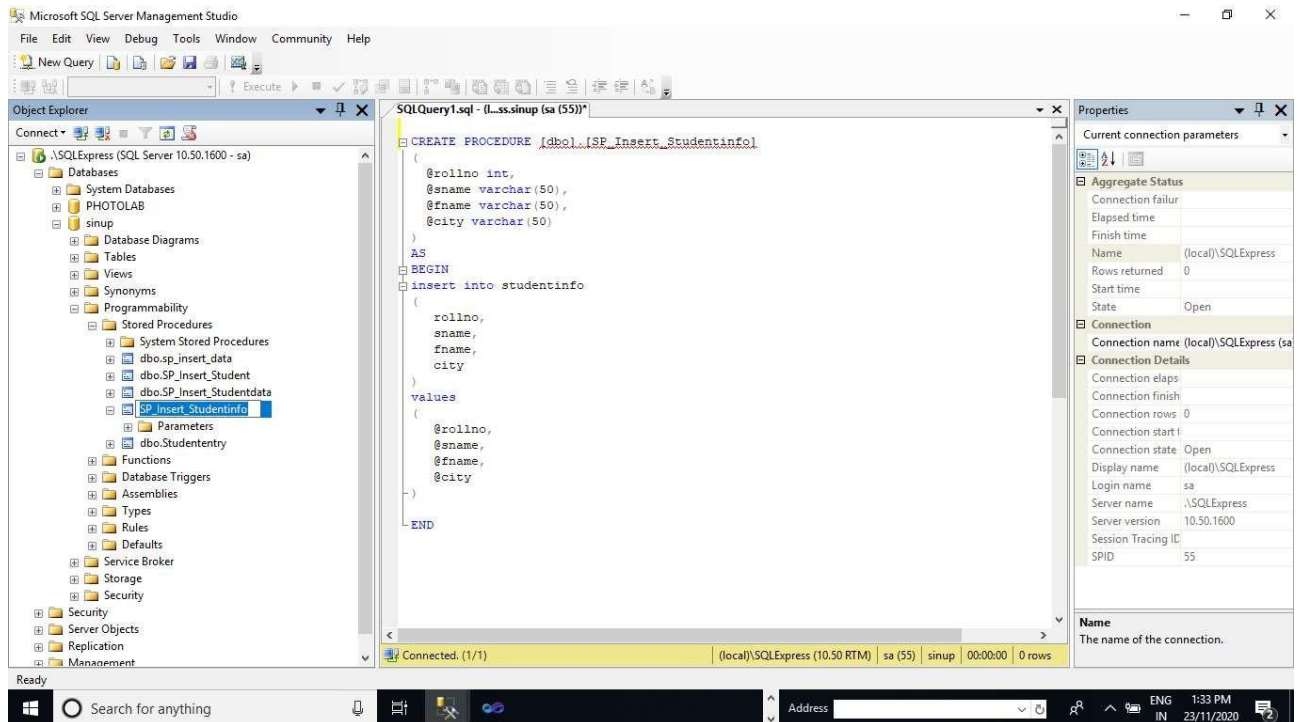
```

Information Form Database:

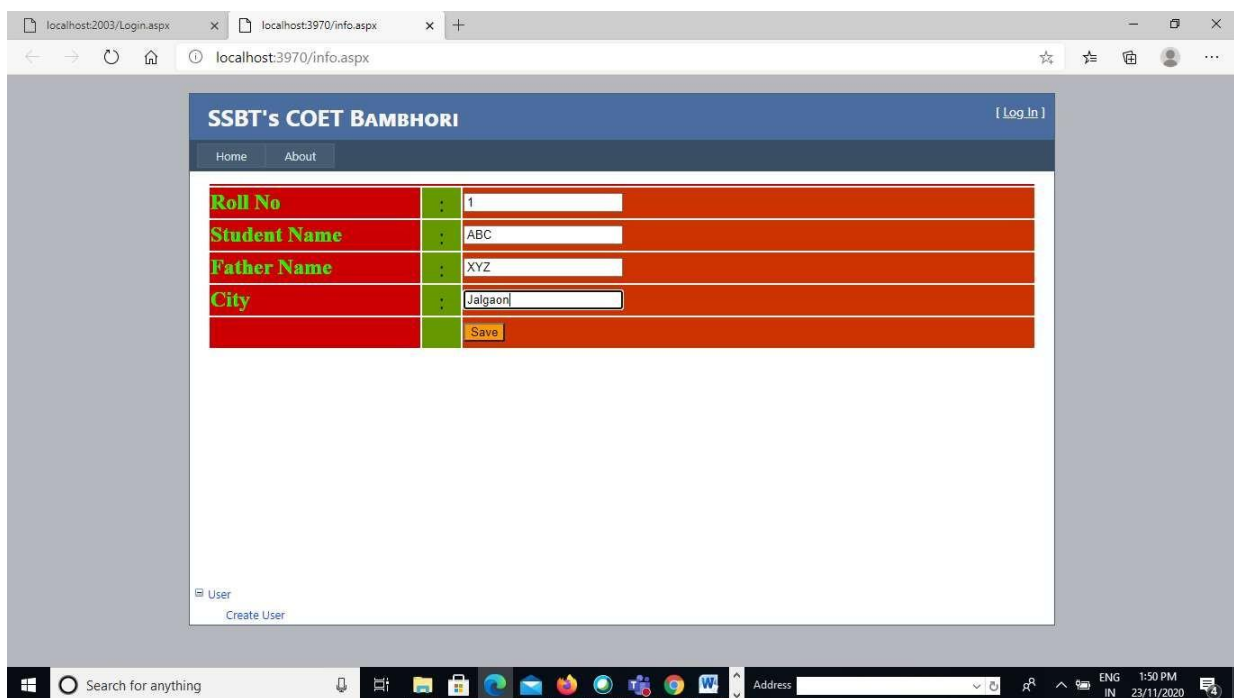
Create Table



Stored Procedure



Output



Report Design

```

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm3.aspx.cs"
Inherits="WebApplication1.WebForm3" %>

<!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></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>

            <asp:GridView ID="gvData" runat="server" AllowPaging="True"
                AutoGenerateColumns="False" CellPadding="4" ForeColor="#333333"
                GridLines="None" onpageindexchanging="gvData_PageIndexChanging" PageSize="2">
                <AlternatingRowStyle BackColor="White" ForeColor="#284775" />
                <Columns>
                    <asp:BoundField DataField="PersonID" HeaderText="PersonID" />
                    <asp:BoundField DataField="LastName" HeaderText="Last Name" />
                    <asp:BoundField DataField="FirstName" HeaderText="First Name" />
                    <asp:BoundField DataField="Address" HeaderText="Address" />
                    <asp:BoundField DataField="City" HeaderText="City" />
                    <asp:CommandField ShowEditButton="true" />
                    <asp:CommandField ShowDeleteButton="true" />
                </Columns>
                <EditRowStyle BackColor="#999999" />
                <FooterStyle BackColor="#5D7B9D" Font-Bold="True" ForeColor="White" />
                <HeaderStyle BackColor="#5D7B9D" Font-Bold="True" ForeColor="White" />
                <PagerStyle BackColor="#284775" ForeColor="White" HorizontalAlign="Center" />
                <RowStyle BackColor="#F7F6F3" ForeColor="#333333" />
                <SelectedRowStyle BackColor="#E2DED6" Font-Bold="True" ForeColor="#333333" />
                <SortedAscendingCellStyle BackColor="#E9E7E2" />
                <SortedAscendingHeaderStyle BackColor="#506C8C" />
                <SortedDescendingCellStyle BackColor="#FFFDF8" />
                <SortedDescendingHeaderStyle BackColor="#6F8DAE" />
            </asp:GridView>

        </div>
    </form>
</body>
</html>

```

Report Asp.net Code

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;
namespace WebApplication1
{
    public partial class WebForm3 : System.Web.UI.Page

```

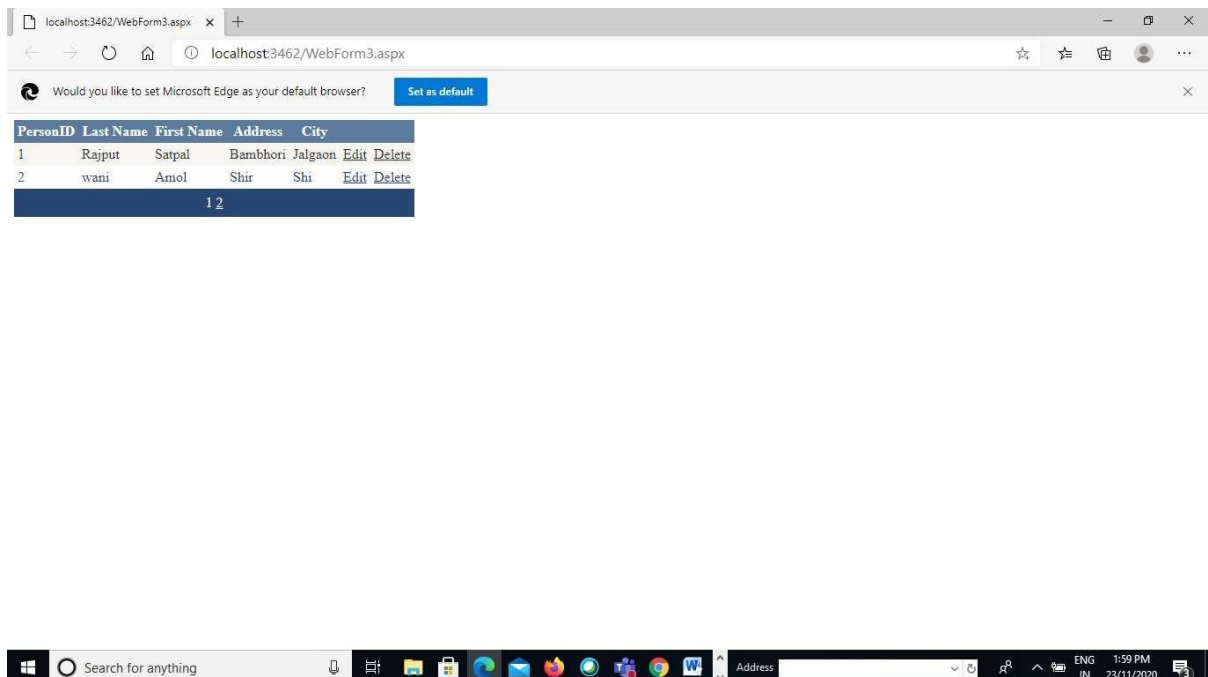
```

{
    SqlConnection con = new SqlConnection("Data Source=.\SQLExpress;Initial
Catalog=sinup;User ID=sa;Password=ssbt");
    protected void Page_Load(object sender, EventArgs e)
    {
        con.Open();
        show();
    }
    private void show()
    {
        SqlCommand cmd = new SqlCommand("select * from Persons", con);
        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataSet ds = new DataSet();
        da.Fill(ds);
        gvData.DataSource = ds;
        gvData.DataBind();
    }

    protected void gvData_PageIndexChanging(object sender, GridViewPageEventArgs e)
    {
        gvData.PageIndex = e.NewPageIndex;
        show();
    }
}
}

```

Output



5. Post-lab Task:

Outcomes:

1. Creates ASP.net Web Forms
2. Created ADO.net programs that use various library functions, and that manipulate database.

Questions and Answers:

Questions: 1. What type of Variables uses in ASP.NET?

Questions: 2. Explain Datatype uses in ASP.NET?

Questions: 3. How Database Connectivity done in ASP.NET?

Handwritten Answer on Next Blank Page of same margin



**GANGAMAI COLLEGE OF ENGINEERING, NAGAON.
DEPARTMENT OF COMPUTER**

Name: _____

Class: B.E. Computer

Division :

Batch:

Roll No:

Subject: **Advance Technology Lab II**

Date of Performance: __/__/20__

Date of Completion: __/__/20__

Grade:

Sign:

Experiment No. 3

Aim: Create a Ruby on Rails an application (Use Technology)

1. Objective:

2. Steps to develop an application:

Install Ruby On Rails on Ubuntu

The first step is to install some dependencies for Ruby and Rails.

To make sure we have everything necessary for Webpacker support in Rails, we're first going to start by adding the Node.js and Yarn repositories to our system before installing them.

```
$sudo apt install curl
```

```
$sudo apt-get update
```

```
$sudo apt-get install git-core zlib1g-dev build-essential libssl-dev libreadline-dev libyaml-dev  
libsqlite3-dev sqlite3 libxml2-dev libxslt1-dev libcurl4-openssl-dev software-properties-common  
libffi-dev nodejs yarn
```

Installing with rbenv is a simple two step process. First you install rbenv, and then ruby-build:

```
cd
```

```
git clone https://github.com/rbenv/rbenv.git ~/.rbenv
```

```
echo 'export PATH="$HOME/.rbenv/bin:$PATH"' >> ~/.bashrc
```

```
echo 'eval "$(rbenv init -)"' >> ~/.bashrc
```

```
exec $SHELL
```

```
2. bash
$
[16:38:16][~/Documents/Projects/rails/todo_list][ruby-2.4.1][node-7.9.0][master *]
$ rake
Run options: --seed 39324

# Running:

.....

Finished in 0.853131s, 11.7215 runs/s, 16.4101 assertions/s.
10 runs, 14 assertions, 0 failures, 0 errors, 0 skips
[16:39:03][~/Documents/Projects/rails/todo_list][ruby-2.4.1][node-7.9.0][master *]
$
```

The screenshot shows a web browser window with the title 'ToDoList' and a single tab. The address bar displays 'localhost:3000/tasks/9/edit'. The page has a dark header with the text 'Editing Book conference ticket' and a green 'Go back' button. The main content area is white and contains a form with the following fields:

- Task:** A text input field containing 'Book conference ticket' with a small menu icon on the right.
- Details:** A larger text input field containing 'Also book flight and hotel...'.
- Due date:** A text input field containing '08/12/2017'.
- Completed:** A label followed by an unchecked checkbox.

At the bottom of the form is a green 'Update Task' button.

git clone

The screenshot shows a web browser window with the title 'ToDoList' and the address bar displaying 'localhost:3000/tasks'. The page has a dark header with the text 'New task' and a green 'Go back' button. A prominent red error banner contains the message '2 errors prohibited this task from being saved:' followed by a bulleted list of errors: 'Task can't be blank' and 'Task is too short (minimum is 3 characters)'. Below the banner, the form consists of three input fields: 'Task' (empty), 'Details' (empty), and 'Due date' (containing the placeholder 'mm/dd/yyyy'). There is also a 'Completed' checkbox which is currently unchecked. At the bottom of the form is a green 'Create Task' button.

ToDoList x Blanca

localhost:3000/tasks

New task

Go back

2 errors prohibited this task from being saved:

- Task can't be blank
- Task is too short (minimum is 3 characters)

Task

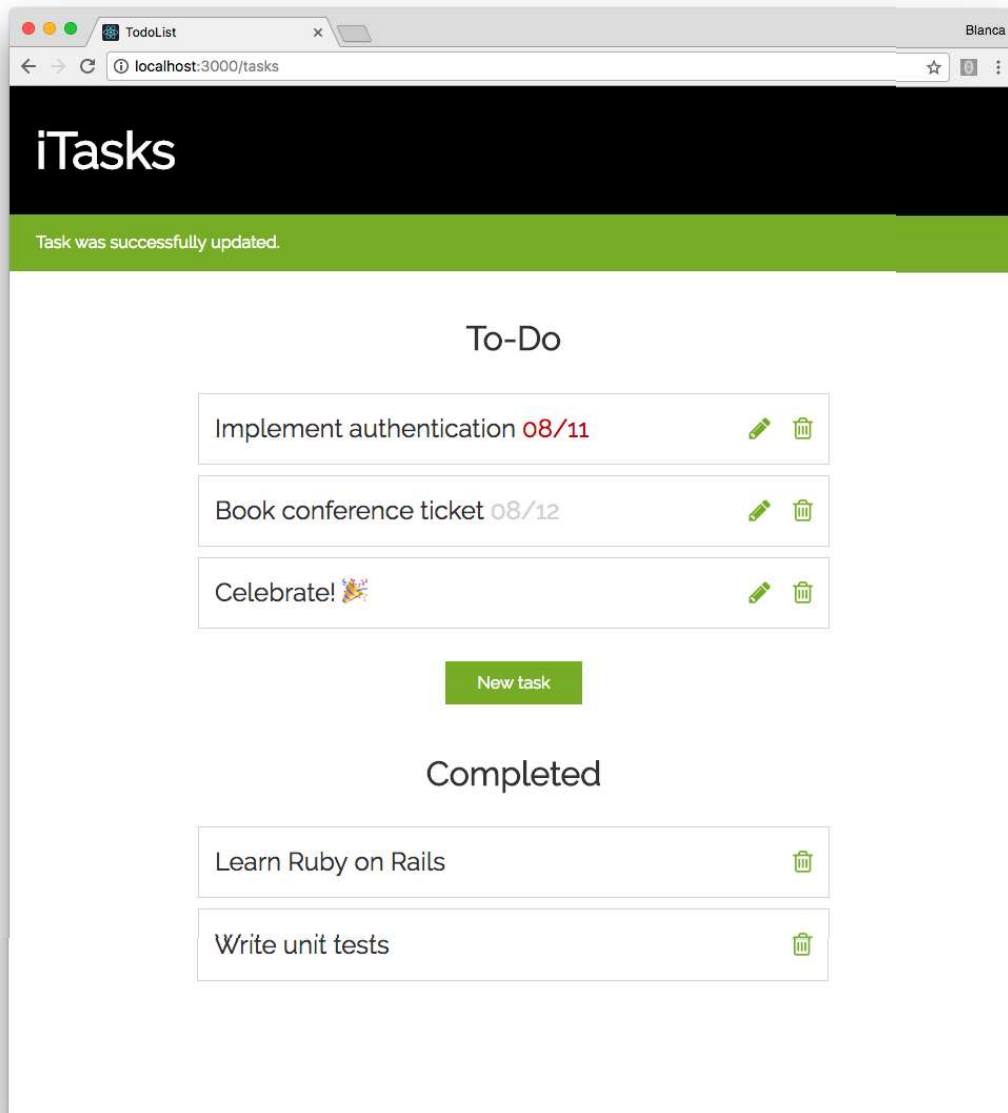
Details

Due date

Completed ☐

Create Task

OUTPUT:



Functionality:

- As a user, I can add a task to the list.
- As a user, I can see all the tasks on the list in an overview.
- As a user, I can drill into a task to see more information about the task.
- As a user, I can delete a task.
- As a user, I can mark a task as completed.
- As a user, when I see all the tasks in the overview, if today's date is past the task's deadline, highlight it.

```

        <%= form.submit class: 'action-button' %>
      </div>
    <% end %>
  </div>
<% end %>

```

Application.html.erb

```

<!DOCTYPE html>
<html>
  <head>
    <title>iTasks</title>
    <%= csrf_meta_tags %>

    <%= stylesheet_link_tag      'application',
'https://fonts.googleapis.com/css?family=Raleway', media: 'all', 'data-
turbolinks-track': 'reload' %>
    <%= javascript_include_tag 'application', 'data-turbolinks-track':
'reload' %>
  </head>

  <body>
    <%= yield %>
  </body>
</html>

```

Mailer.html.erb

```

<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <style>
      /* Email styles need to be inline */
    </style>
  </head>

  <body>
    <%= yield %>
  </body>
</html>

```



```

        <% end %>
    </ul>
</div>
<% end %>

<div class="form-container">
    <div class="form-field">
        <div class="form-label-container">
            <%= form.label :task, class: 'form-label' %>
        </div>
        <div class="form-input-container">
            <%= form.text_field :task, id: :task_task, class: 'form-input',
disabled: readonly %>
        </div>
    </div>

    <div class="field form-field">
        <div class="form-label-container">
            <%= form.label :details, class: 'form-label' %>
        </div>
        <div class="form-input-container">
            <%= form.text_area :details, id: :task_details, class: 'form-input',
disabled: readonly %>
        </div>
    </div>

    <div class="form-field">
        <div class="form-label-container">
            <%= form.label :due_date, class: 'form-label' %>
        </div>
        <div class="form-input-container">
            <%= form.date_field :due_date, id: :task_due_date, class: 'form-
input', disabled: readonly %>
        </div>
    </div>

    <div class="field form-field">
        <div class="form-label-container">
            <%= form.label :completed, class: 'form-label' %>
        </div>
        <div class="form-input-container">
            <%= form.check_box :completed, id: :task_completed, class: 'form-
input', disabled: readonly %>
        </div>
    </div>

    <% unless readonly %>
        <div class="field form-buttons">

```

```

    <% end %>
  </ul>
</div>
</div>

```

task.html.erb

```

<%= link_to task, class: 'task-name' do %>
  <%= task.task %>
  <% if task.due_date.present? %>
    <time class="task-time <%= task.due_date <= Date.today ? 'is-due' : '' %>"
datetime="<%= task.due_date.strftime('%FT%T') %>">
      <%= task.due_date.strftime("%m/%d") %>
    </time>
  <% end %>
<% end %>

```

Edit.html.erb

```

<h1 class="task-header">
  Editing <%= @task.task %>
</h1>

<%= link_to 'Go back', tasks_path, class: ['action-button', 'back-button'] %>

<%= link_to 'Log out', destroy_user_session_path, method: :delete, class:
['action-button', 'log-out-button'] %> ⓘ

<%= render 'form', task: @task, readonly: false %>

```

form.html.erb

```


<%= form_with(model: task, local: true) do |form| %>
  <% if task.errors.any? %>
    <div class="errors-container">
      <h2>
        <%= pluralize(task.errors.count, "error") %> prohibited this task from
being saved:
      </h2>

      <ul class="errors-list">
        <% task.errors.full_messages.each do |message| %>
          <li class="errors-item">
            <%= message %>
          </li>
        <% end %>
      </ul>
    </div>
  <% end %>

  <%= form.text_field :task, :readonly => task.task == task.task %>
  <%= form.submit %>
</form>

```

```

<%= link_to 'Log out', destroy_user_session_path, method: :delete, class:
['action-button', 'log-out-button'] %> 

<div class="tasks-container">
  <div class="tasks-todo">
    <h2 class="tasks-status">
      To-Do
    </h2>

    <ul class="tasks-list">
      <% @tasks.todo.each do |task| %>
        <% if task.user == current_user %>
          <li class="tasks-item">
            <%= render 'task', task: task %>
            <div class="task-buttons">
              <%= link_to fa_icon("pencil"), edit_task_path(task), class:
'task-button' %>
              <%= link_to fa_icon("trash-o"), task, class: 'task-button',
method: :delete, data: { confirm: 'Are you sure?' }
%>
            </div>
          </li>
        <% end %>
      <% end %>
    </ul>
  </div>

  <div class="tasks-buttons">
    <%= link_to 'New task', new_task_path, class: "action-button" %>
  </div>

  <div class="tasks-completed">
    <h2 class="tasks-status">
      Completed
    </h2>

    <ul class="tasks-list">
      <% @tasks.completed.each do |task| %>
        <% if task.user == current_user %>
          <li class="tasks-item">
            <%= render 'task', task: task %>
            <div class="task-buttons">
              <%= link_to fa_icon("trash-o"), task, class: 'task-button',
method: :delete, data: { confirm: 'Are you sure?' }
%>
            </div>
          </li>
        <% end %>
      <% end %>
    </ul>
  </div>
</div>

```

todo-list/app/models/user.rb

```
class User < ApplicationRecord
  # Include default devise modules. Others available are:
  # :confirmable, :lockable, :timeoutable and :omniauthable
  devise :database_authenticatable, :registerable,
         :recoverable, :rememberable, :trackable, :validatable

  has_many :tasks
end
```

New.html.erb

```
<h1 class="task-header">
  New task
</h1>

<%= link_to 'Go back', tasks_path, class: ['action-button', 'back-button'] %>

<%= link_to 'Log out', destroy_user_session_path, method: :delete, class:
['action-button', 'log-out-button'] %>

<%= render 'form', task: @task, readonly: false %>
```

Show.html.erb

```
<h1 class="task-header">
  <%= @task.task %>
</h1>

<%= link_to 'Go back', tasks_path, class: ['action-button', 'back-button'] %>

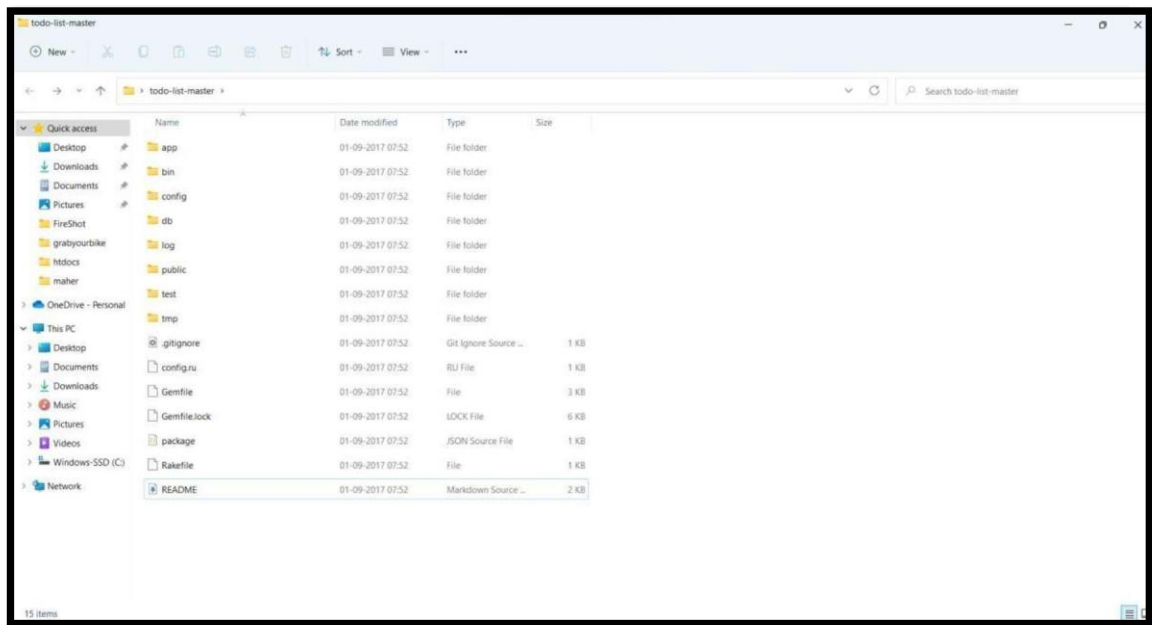
<%= render 'form', task: @task, readonly: true %>
```

Index.html.erb

```
<h1 class="tasks-header">
  iTasks
</h1>

<p class="notice-container"><%= notice %></p>
```

Project Directory:



todo-list/app/models/application_record.rb

```
class ApplicationRecord < ActiveRecord::Base
  self.abstract_class = true
end
```

todo-list/app/models/task.rb

```
class Task < ApplicationRecord
  belongs_to :user

  validates :task, presence: true,
                 length: { minimum: 3 }

  scope :completed, -> {
    where(:completed => true)
  }

  scope :todo, -> {
    where(:completed => false)
  }
end
```

```
# If you would like to set a password for the user, you can do the following
sudo -u postgres psql
postgres=# \password chris
```

Final Steps

And now for the moment of truth. Let's create your first Rails application: ##### If you want to use SQLite (not recommended)

```
rails new myapp
```

If you want to use MySQL rails new

```
myapp -d mysql
```

If you want to use Postgres

Note that this will expect a postgres user with the same username

as your app, you may need to edit config/database.yml to match the# user you created earlier

```
rails new myapp -d postgresql
```

Move into the application directory

```
myapp
```

If you setup MySQL or Postgres with a username/password, modify the

config/database.yml file to contain the username/password that you specified

Create the database

```
rake db:create rails
```

```
server
```

You can now visit <http://localhost:3000> to view your new website!

Now that you've got your machine setup, it's time to start building some Rails applications.

If you received an error that said Access denied for user 'root'@'localhost' (using password: NO) then you need to update your config/database.yml file to match the database username and password.

If you get a different result for some reason, it means your environment may not be setup properly.

Setting Up A Database

Rails ships with `sqlite3` as the default database. Chances are you won't want to use it because it's stored as a simple file on disk.

If you're new to Ruby on Rails or databases in general, I strongly recommend setting up PostgreSQL.

If you're coming from PHP, you may already be familiar with MySQL.

Setting Up MySQL

Rails ships with `sqlite3` as the default database. Chances are you won't want to use it because it's stored as a simple file on disk.

```
sudo apt-get install mysql-server mysql-client libmysqlclient-dev
```

Installing the `libmysqlclient-dev` gives you the necessary files to compile the `mysql2` gem which is what Rails will use to connect to MySQL when you setup your Rails app.

Setting Up PostgreSQL

For PostgreSQL, we're going to add a new repository to easily install a recent version of Postgres.

```
sudo apt install postgresql-11 libpq-dev
```

The postgres installation doesn't setup a user for you, so you'll need to follow these steps to create a user with permission to create databases. Feel free to replace `chris` with your username.

```
sudo -u postgres createuser chris -s
```

```
git clone https://github.com/rbenv/ruby-build ~/.rbenv/plugins/ruby-build
echo 'export PATH="$HOME/.rbenv/plugins/ruby-build/bin:$PATH"' >> ~/.bashrc exec
$SHELL
```

```
rbenv install 3.0.1
rbenv global 3.0.1 ruby
v
```

The last step is to install Bundler

```
gem install bundler
```

Installing Rails

Choose the version of Rails you want to install:

6.1.3.2 (Recommended)

```
gem install rails -v 6.1.3.2
```

If you're using rbenv, you'll need to run the following command to make the rails executable available:

```
rbenv rehash
```

Now that you've installed Rails, you can run the rails -v command to make sure you have everything installed correctly:

```
rails -v
# Rails 6.1.3.2
```


Outcomes:

Able to deploy project in Ruby on rail

Questions and Answers:

Questions: 1. What type of Variables uses in RubyRail?

Questions: 1. Explain Datatype uses in RubyRail?

Questions: 1. How Database Connectivity done in RubyRail?

Handwritten Answer on Next Blank Page of same margin