Fall 2015

GSU Alumni Portal

Vemuri Vinusha Chowdary
Governors State University

Sairam Dasu
Governors State University

Siva Machineni
Governors State University

Follow this and additional works at: http://opus.govst.edu/capstones
Part of the Computer and Systems Architecture Commons, and the Computer Sciences Commons

Recommended Citation
Chowdary, Vemuri Vinusha; Dasu, Sairam; and Machineni, Siva, "GSU Alumni Portal" (2015). All Capstone Projects. 163.
http://opus.govst.edu/capstones/163

For more information about the academic degree, extended learning, and certificate programs of Governors State University, go to http://www.govst.edu/Academics/Degree_Programs_and_Certifications/

Visit the Governors State Computer Science Department
This Project Summary is brought to you for free and open access by the Student Capstone Projects at OPUS Open Portal to University Scholarship. It has been accepted for inclusion in All Capstone Projects by an authorized administrator of OPUS Open Portal to University Scholarship. For more information, please contact opus@govst.edu.
Table of Contents

1 Project Description ................................................................................................................................. 1
  1.1 Project Abstract ............................................................................................................................... 1
  1.2 Competitive Information ................................................................................................................. 1
  1.3 Relationship to Other Applications/Projects ................................................................................... 1
  1.4 Future Enhancements ...................................................................................................................... 1
  1.5 Definitions and Acronyms .............................................................................................................. 1
2 Technical Description ............................................................................................................................ 2
  2.1 Project/Application Architecture ..................................................................................................... 2
  2.2 Project/Application Information flows ............................................................................................ 3
  2.3 Capabilities ...................................................................................................................................... 4
  2.4 Risk Assessment and Management .................................................................................................. 4
3 Project Requirements ............................................................................................................................. 4
  3.1 Identification of Requirements ........................................................................................................ 4
4 Project Design Description .................................................................................................................... 5
5 Project Design Units Impacts .................................................................................................................. 17
  5.1 Functional Area/Design Unit A ..................................................................................................... 22
    5.1.1 Functional Overview ................................................................................................................. 22
6 Conclusion ............................................................................................................................................ 29
7 Acknowledgements ............................................................................................................................... 30
8 References ............................................................................................................................................. 31
9 Appendix ............................................................................................................................................... 32
1  Project Description

1.1  Project Abstract

The main objective of Gsu alumni portal web application is to allow old and new students of a university to communicate with each other. The application allows students to register and then search the data based on different criteria. Also it has the benefit of having a centralized database and up to date information. A user can easily obtain information about other registered users. This application is deployed using Cloud computing (Microsoft Azure) and nothing is managed locally. The administrator is responsible for maintaining information of students. When a student submits the registration form, administrator will complete the verification process and, if successful, the student details are added into the database. The administrator maintains the passwords of Event Manager and that of himself. Event Manager Module maintains the information about various events organized by the university. Details of notifications are also maintained. Event manager can add, edit and view event details. And event manager sends an SMS about the events to the registered students and Administrator. The Students after successful registration process, they can logon into their account and can send mails, post queries, update their profiles and even search for other student details. One can view the event details and search for specific information. The module provides mail and query functionalities.

1.2  Competitive Information

The GSU Alumni portal web application is deployed using Cloud computing, Microsoft azure as a platform. We can overcome single point Hardware failure by using cloud computing. The other pros of cloud computing are no hardware to maintain, unlimited instance scaling, unlimited disk scaling, Dynamic scaling, pay for what we use, Resilient and Redundant.

1.3  Relationship to Other Applications/Projects

The GSU Alumni portal Db is related to MSSQL, Views are related to visual studio using Asp.net technology. The local Db is connected to SQL Azure (it can be managed in cloud too). The local Asp.net project is published in cloud using windows azure as a platform.

1.4  Future Enhancements

For Future Enhancements the GSU alumni portal can have an automated e-mail, when a new event gets updated. Furthermore admin can have an automated e-mail, whenever a new student registers in the Portal instead of constantly checking the student Registrations.

1.5  Definitions and Acronyms

- SSMS - Sql Server Management Studio.
- SQL - Structured Query Language.
- SDK - Software Development Kit.
- SMTP - Simple Mail Protocol.
2 Technical Description

2.1 Project/Application Architecture

- While Developing the GSU Alumni Portal web application the below mentioned architecture is implemented using Asp.net Technology.

![Asp.net web application architecture](image)

Figure 1 (Asp.net web application architecture)
The asp.net web application is deployed using Microsoft windows azure. After the completion of the business logic using Asp.net, the application is deployed by using the following architecture.

Figure 2 (3- Tier Azure Deployment)

2.2 *Project/Application Information flows*

The application information flows through the Administrator, Event Manager and students Modules.

Figure 3 (Sequence Diagram)
2.3 **Capabilities**

The Database application provides capabilities to support business application such as retrieving/adding/deleting/updating user data. The connection string should be placed appropriately between database layer and Asp.net web application while developing the Project.

2.4 **Risk Assessment and Management**

The risks involved in this project are unsupportive versions of software, while publishing the web application to cloud using Microsoft windows azure. The risks can be managed by choosing the compatible software requirements.

3 **Project Requirements**

3.1 **Identification of Requirements**

This Project is mainly divided in to three modules Administrator, Event Manager and student. To implement these modules successfully the following project requirements are maintained in this project.

- **Software Requirements:**
  2. SQL Server Management studio 2015.
  3. Microsoft .Net Frame work 4.0(Minimum).
  4. Azure SDK.

- **Hardware Requirements:**
  1. RAM: 8GB.
  2. Available Hard Disk Space: 40GB.

- **Web hosting Requirement:**
  1. Microsoft windows pay-as-you-go account.
4 Project Design Description

• Database Design

The project is designed using asp.net as front end and sql server database as backend. The sql database is managed in the cloud by using widows azure sql Databases. The database is designed using Tables and stored procedures. The database can be managed in the cloud by using by following URL.

URL: https://aluminiserver.database.windows.net/?langid=en-us#$database=AluminiPoralGradSeminar

This url directs to full functionalities of the database, where the database can be managed in cloud with the valid server credentials. As discussed the database the project is designed using Tables and stored procedures. The complete view of database design is provided the below screen shots. The database design is explained using Dependency diagrams.

• Database Login:

![Microsoft Azure](image)

**SQL DATABASE**

**SERVER**

aluminiserver.database.windows.net

**DATABASE**

AluminiPoralGradSeminar

**USERNAME**

vinusha

**PASSWORD**

********

[Log on] [Cancel]

Figure 4 (Azure Sql Database Login)
After log on in to Sql database server, we can view the existing data, queried and database data can be changed according to the requirement.

![AluminiPoratGradSeminar](image)

Figure 5 (Azure Sql Database name)

![aluminisher.database.windows.net > [AluminiPoratGradSeminar] > Tables](image)

Figure 6 (Database Parameters – List of Tables)
Figure 7 (List of Stored Procedures)

This figure illustrates the all the dependencies.
Front End Design:

The front end pages are designed using Asp.net. The code behind the business logic is discussed with the design view below. The main code functionalities are in Home, Dashboard, Event dashboard and Student dashboard.

Home.aspx: The code in Home.aspx page mainly deals with login credentials for administrator, event manager and student. The student registrations are also included in home.aspx.

Code behind business logic:

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using AlumniPortal.BO;

namespace AlumniPortal
{
    public partial class Login : System.Web.UI.Page
    {
        ...
    }
}
```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        BO.SessionManager.SessionInfo.student = null;
    }
}

protected void btnRegister_Click(object sender, EventArgs e)
{
    try
    {
        StudntRegBO objstudentDetails = new StudntRegBO
        {
            studentId = txtstudentid.Value,
            student_Fname = fname.Value,
            student_Lname = lname.Value,
            gender = gender.Value,
            graduate_year = graduateYear.Value,
            email = email.Value,
            phone = phone.Value,
            address = address.Value,
            username = username.Value,
            password = password.Value,
            role = "student",
            Company = company.Value
        };
        DAL objDal = new DAL();
        int result = objDal.studentRegistration(objstudentDetails);
        ClientScript.RegisterStartupScript(GetType(), "alert", "alert('Registration Sucessfull,
Kindly wait for Admin Approval');", true);
    }
    catch (Exception ex)
    {
        throw ex;
    }
}

protected void btnSubmit_Click(object sender, EventArgs e)
{
    DAL objDal = new DAL();
    var userDetails = objDal.validateUser(Lusername.Value, Lpassword.Value);
    if (userDetails != null)
    {
        // Further code...
    }
}
BO.SessionManager.SessionInfo.student = userDetails;
if (BO.SessionManager.SessionInfo.student.role == "admin")
{
    Response.Redirect("Dashboard.aspx");
}
else if (BO.SessionManager.SessionInfo.student.role == "event")
{
    Response.Redirect("eventDashboard.aspx");
}
else if (BO.SessionManager.SessionInfo.student.role == "student")
{
    Response.Redirect("studDashboard.aspx");
}
else
{
    ClientScript.RegisterStartupScript(GetType(), "alert", "alert('Invalied Credentials');", true);
}

Design view of Home.aspx:

![Design view of Home.aspx](image)

Figure 9 (Design view of Home.aspx)
Dashboard.aspx: The code in the Dashboard.aspx mainly deals with the administrator functionalities.

Code behind business logic:

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using AlumniPortal.BO;

namespace AlumniPortal
{
    public partial class Dashboard : System.Web.UI.Page
    {
        DAL objDal = new DAL();
        protected void Page_Load(object sender, EventArgs e)
        {
            Response.Cache.SetNoStore();
            if (BO.SessionManager.SessionInfo.student != null)
            {
                if (!IsPostBack)
                {
                    gvbind();
                }
            }
            else
            {
                Response.Redirect("Login.aspx");
            }
        }
        public void gvbind()
        {
            grdDetails.DataSource = objDal.getStudentDetails(""),
            grdDetails.DataBind();
            grdView.DataSource = objDal.geteventDetails();
            grdView.DataBind();
        }
        protected void grdDetails_RowUpdating(object sender, GridViewCommandEventArgs e)
        {
            if (e.CommandName == "Approve")
            {
                int index = Convert.ToInt32(e.CommandArgument.ToString());
                string id = grdDetails.Rows[index].Cells[0].Text.ToString();
            }
        }
    }
}
```
```csharp
    string name = grdDetails.Rows[index].Cells[1].Text.ToString();
    DAL objDal = new DAL();
    objDal.approveStudent(id.ToString(), name.ToString());
    gvbind();
}

protected void grdView_RowEditing(object sender, GridViewEditEventArgs e)
{
    grdView.EditIndex = e.NewEditIndex;
    gvbind();
}

protected void grdView_RowUpdating(object sender, GridViewUpdateEventArgs e)
{
    int userid = Convert.ToInt32(grdView.DataKeys[e.RowIndex].Value.ToString());
   GridViewRow row = (GridViewRow)grdView.Rows[e.RowIndex];

eventBo objeventBo = new eventBo {
    eventId = userid,
    name = ((TextBox)row.Cells[1].Controls[0]).Text,
    date = ((TextBox)row.Cells[2].Controls[0]).Text.ToString(),
    time = ((TextBox)row.Cells[3].Controls[0]).Text.ToString(),
    description = ((TextBox)row.Cells[4].Controls[0]).Text.ToString(),
    venue = ((TextBox)row.Cells[5].Controls[0]).Text.ToString(),
    number = ((TextBox)row.Cells[6].Controls[0]).Text.ToString()
};
    objDal.updateevent(objeventBo);
    gvbind();
}

protected void grdView_RowCancelingEdit(object sender, GridViewCancelEditEventArgs e)
{
    grdView.EditIndex = -1;
    gvbind();
}
```
Eventdashboard.aspx: The code in eventdashboard.aspx mainly illustrates the Event manager functionalities.

Code behind Business logic:

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using AlumniPortal.BO;
using System.Data;
using Twilio;

namespace AlumniPortal
{
    partial class eventDashboard : System.Web.UI.Page
    {
        DAL objDal = new DAL();
        protected void Page_Load(object sender, EventArgs e)
        {
            Response.Cache.SetNoStore();
            if (BO.SessionManager.SessionInfo.student != null)
```
if (!IsPostBack)
{
    gvbind();
}
}
else
{
    Response.Redirect("Login.aspx");
}

public void gvbind()
{
    grdView.DataSource = objDal.geteventDetails();
grdView.DataBind();
}

protected void grdView_RowEditing(object sender, GridViewEditEventArgs e)
{
    grdView.EditIndex = e.NewEditIndex;
gvbind();
}

protected void grdView_RowUpdating(object sender, GridViewUpdateEventArgs e)
{
    int userid = Convert.ToInt32(grdView.DataKeys[e.RowIndex].Value.ToString());
    GridViewRow row = (GridViewRow)grdView.Rows[e.RowIndex];

    eventBo objeventBo = new eventBo {
        eventId=userid,
        name = ((TextBox)row.Cells[0].Controls[0]).Text,
        date = ((TextBox)row.Cells[1].Controls[0]).Text.ToString(),
        time =((TextBox)row.Cells[2].Controls[0]).Text.ToString(),
        description =((TextBox)row.Cells[3].Controls[0]).Text.ToString(),
        venue = ((TextBox)row.Cells[4].Controls[0]).Text.ToString(),
        number = ((TextBox)row.Cells[5].Controls[0]).Text.ToString()
    };
    objDal.updateevent(objeventBo);
gvbind();
}

protected void grdView_RowCancelingEdit(object sender, GridViewCancelEditEventArgs e)
{
    grdView.EditIndex = -1;
gvbind();
}
protected void Calendar1_SelectionChanged(object sender, EventArgs e)
{
    string dateselectd = Calendar1.SelectedDate.Date.ToString("MM/dd/yyyy");
    string eventDescription="No Events";
    string eventdate="";
    eventdesbox.Text = "";
    DataSet ds = new DataSet();
    ds = objDal.geteventDetails();
    for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
    {
        eventDescription = ds.Tables[0].Rows[i]["description"].ToString();
        eventdate = ds.Tables[0].Rows[i]["date"].ToString();
        if (dateselectd == eventdate)
        {
            eventdesbox.Text = eventDescription.ToString();
        }
    }
    if(eventdesbox.Text.Trim()=="")
    {
        eventdesbox.Text = "Whoops! No events found";
    }
}

protected void Button1_Click(object sender, EventArgs e)
{
    try
    {
        // Find your Account Sid and Auth Token at twilio.com/user/account
        string AccountSid = "AC308c41d8ca8b60924ea3deb2e3489e69";
        string AuthToken = "85bacc582d60e582757e2433ea8f08c1";
        string Number = "";

        DAL objDal = new DAL();
        DataTable ds = new DataTable();
        ds = objDal.getStudentDetails("").Tables[0];
        for (int i = 0; i < ds.Rows.Count; i++)
        {
            Number = ds.Rows[i]["phone"].ToString();
        
            TwilioRestClient client;
            client = new TwilioRestClient(AccountSid, AuthToken);
        }
    }
}
Message result = client.SendMessage("+16306353090", Number, eventdesbox.Text);

    if (result.RestException != null)
        {
            // an exception occurred making the REST call
            string message = result.RestException.Message;
        }
    
    
    catch (Exception)
        {
            throw;
        }
    
Design View:

Figure 11 (Design view of EventDashboard.aspx)
5  Project Design Units Impacts

After successful azure sql database design and asp.net front end pages, by using webapps in windows azure both sql azure database and asp.net web pages are connected. Publish is profile is created in windows azure in order to establish a connection between azure sql databases and windows azure web apps. For proper functioning of Gsu alumni portal web app publishing the webapp from visual studio is important.

Figure 12(Publish profile)
When visual studio is connected to windows azure, the azure sdk automatically generates a publish profile.

Figure 13(Publishing of Web app)

Azure sql database connection string is mentioned in web.config. During publishing the web app in cloud, the cloud database is connected instead of local databases.
The published GSU alumni portal web app destination URL is http://aluminiportal.azurewebsites.net/login.aspx.

In Microsoft Azure portal we can view all the services used to host the web app in cloud.
Figure 14: (Microsoft azure Services used to host a web application)
The usage overview of Gsu alumni portal web app.

5.1 Functional Area/Design Unit A

5.1.1 Functional Overview

After successful deployment, the Gsu alumni portal web app functional view is described as:
Login: This general login web page for Administrator, Event manager and student.

Figure 16(General login Page)

Administrator: The Administrator logins with his/her login user id and password.

Figure 17(Administrator login)

After successful administrator login, he/she can view the existing student, the student waiting for approval, list of events and he can add new employee (administrator or event Manager).
Figure 18 (Administrator can approve or reject the student registrations)
a) Add Employee:

- Event Manager: The Event Manager Logins with his/her user id and password, event manager can add new event and send event notification sms to all registered users in the Gsu alumni portal.

Figure 19(Administrator adds an employee)

Figure 20(Event Manager sends an event sms)
a) Add Event:

![Add Event](image)

Figure 21(Event manager adds an event)

- Student: After successful registration and acceptance from administrator, student can login into Gsu alumni portal and he/she can view the contact information about other registered students, list of events conducted by the university, Email other students and ask queries and can get answers from the administrator.

Student login:

![Student Login](image)

Figure 22(Student Registration)
Figure 23(Student Login)

Figure 24(Gsu alumni portal student view)
• Edit Profile:

![Edit Profile](image)

Figure 25 (Edit Profile)

• Ask admin:

![Ask admin](image)

Figure 26 (Sends queries to admin)
- Email:

  ![Email Form](image)

  **Figure 27**(Email other students)
6 Conclusion

While developing Gsu alumni portal, we have gathered immense knowledge about Asp.net technology and cloud computing. We have understood how a web application works and how can it be made as a user friendly application. It also provides knowledge about the latest technology used in developing applications and client server technology that will have great demand in future. This will provide us better opportunities and guidance in future in developing projects independently.
7 Acknowledgements

It gives me immense pleasure to express my deep sense of gratitude to Professor Dr. Soon-Ok Park, who has been kind enough to guide us in the planning the GSU alumni project. We are very thankful to her for helping us in all possible ways during this project work in field of Computer Science, at Governors State University.

I am also thankful for her continuous feedback and encouragement throughout this project work. Her expertise in various subjects helped me to gain knowledge in various aspects, which will be a great help in my future endeavours.

Finally, I would like to extend thanks to my team members for their continuous support, discussions and suggestions in developing the Gsu alumni project.
8 References

- For installing visual studio 2015

- For installing Sql server 2014

- For creating Asp.net web pages

- To create database in sql azure

- To create web app in windows azure

- To publish web app using visual studio 2015
9 Appendix

List of Figures:

- Figure 1 - Asp.net web application architecture
- Figure 2 - 3- Tier Azure Deployment
- Figure 3 - Sequence Diagram
- Figure 4 - Azure sql database login
- Figure 5 - Azure sql database name
- Figure 6 - Database parameters(list of tables)
- Figure 7 - List of stored procedures
- Figure 8 - Dependencies
- Figure 9 - Design view of home.aspx
- Figure 10 - Design view of Dashboard.aspx
- Figure 11 - Design view of Eventdashboard.aspx
- Figure 12 - Publishing profile
- Figure 13 - Publishing of web app
- Figure 14 - Microsoft azure Services used to host a web application
- Figure 15 - Usage overview
- Figure 16 - General login page
- Figure 17 - Administrator login
- Figure 18 - Administrator can approve or reject the student registrations
- Figure 19 - Administrator adds an employee
- Figure 20 - Event Manager sends an event sms to the registered users in the portal
- Figure 21 - Event manager adds an event
- Figure 22 - Student Registration
- Figure 23 - Student Login
- Figure 24 - Gsu alumni portal student view
- Figure 25 - Edit Profile
- Figure 26 - Sends queries to admin
- Figure 27 - Email other students