CREATION OF STRUCTURED FORMATTED DATABASE FOR ALUMNI PROJECT

Đorđe Dihovični[1]*,
Dragan Kreculj[1],
Nada Ratković Kovačević[1],
Petar Jakovljević[2]

[1]The Academy of Applied Technical Studies Belgrade,
Belgrade, Serbia

[2]Polytechnic-school for new technologies,
Belgrade, Serbia

Abstract:
An Alumni club, made up of former students, has been working at several prestigious Faculties of the University of Belgrade for a long time. Thanks to the contacts through this organization, as well as the help of former students of these Faculties, a large number of newly graduated students received useful advice first-hand, as well as the possibility of employment in successful companies in the country and abroad. The Academy of Applied Technical Studies Belgrade followed this example and formed Alumni Clubs in its departments and sections. In this way, the Alumni Club started working in the Computer-Mechanical Engineering Section. It took time for students who had graduated a long time ago to sign up for the club, but the response of students is getting bigger and bigger, so the need to create a database and appropriate applications emerged. A database in SQL server is created, after which advanced and automated queries for obtaining relevant data are developed in form of stored procedures.

This paper presents two approaches to creating a formatted database in XML language, in order to enable better connection with the most popular programming languages such as Java, ASP.NET, C#, ColdFusion and others. In the first approach, selecting the appropriate button creates an XML document, and in the second approach filling in the text fields in the application, which is developed in the C# programming language, leads to its creation.

Keywords:
Database, XML, Alumni, Programming.

INTRODUCTION

The goal of forming the Alumni organization is to establish and maintain a connection between the Faculty and graduate students, as well as to encourage cooperation between the Faculty, companies and organizations in which they are employed, [1]. Former students can influence the design of the Faculty's development strategy; influence the profiling of new curricula, and assist in marketing and the promotion of the Faculty itself. It should be emphasized the importance of cooperation between the Faculty and companies where former students work, in terms of drawing the attention of potential employers to the employment of future graduates, [2] The task of the Alumni Club of the Computer-Mechanical Engineering Section is to enable the continuation of socialization established during the studies, and to contribute to the strengthening of the profession.

Correspondence:
Đorđe Dihovični

e-mail:
djdihovicni@atssb.edu.rs

DOI: 10.15308/Sinteza-2023-94-99

The other important task is to provide assistance to graduate students, and to establish active and constant cooperation between the Computer-Mechanical Engineering Section on the one hand, and the economy and society on the other.

The Computer-Mechanical Engineering Section wants to establish a permanent relationship with its graduate students, and inform them about all development plans and education programs, and include all those who want to help the development of this Department or Section. In this sense, communication with graduate students is crucial, not only related to their wishes and needs (studying or finding a job), but also their thinking about the study itself (their suggestions and remarks). The Alumni Club of the Computer-Mechanical Engineering Section establishes and maintains a connection between the Department and its students after graduation, connects all former students through socializing and exchange of experiences, develops cooperation between the Department and companies or organizations where former students work, exchanges professional and business information between association members, establishes cooperation with other Alumni organizations from the country and the region, provides advisory support from graduated students and enables the implementation of more effective scientific and research work, connects students and the business community, provides assistance in the employment of association members.

Considering that the number of members of the Alumni Club is growing more and more, it is necessary to automate the entire procedure and create an appropriate application. To develop the application for entering, deleting, updating and searching new members, the C# programming language was chosen [3]. Manipulation with data is much better with stored procedures and advanced queries and an administrator has the opportunity to get valid and specific types of data regarding Alumni members. Implementation of stored procedures for creating the Alumni database is developed and shown in [4]. This paper shows data storage in a text-structured format using the XML language.

2. THE FIRST APPROACH TO CREATING AN ALUMNI XML FILE

Extensible Markup Language (XML) is widely used as an alternative to a database, as well as for storing application configuration information. XML has a similar syntax to the HTML language, [5]. The XML language begins with the declaration <?xml version="1.0"?>, indicating that the document uses XML code, as well as the appropriate text encoding method. An XML document consist of XML elements which consists of an input character, a closing character, and data or so-called child elements between the input and output characters. Any name for the elements can be used, but care should be taken to be case-sensitive, as well as the fact that XML ignores spaces so that the XML file can be written in one line, [6]-[7]. Tabs can be used because spaces between elements can be ignored. XML is case-sensitive, and no special characters can be used in the content.

XML documents must take into account the order of elements and the data type of individual elements. The scope disambiguates the displayed elements by determining which language a particular element belongs to, [8].

XML documents are formed as tree elements. The XML tree starts with the root element and then moves to child elements. To identify the scope, a prefix is introduced by entering a colon and the character that indicates the prefix, [9].

In the .NET framework, there are classes that can read and operate on XML documents. These classes are located in the System.Xml namespace, [10]. The model used is the XML Document Object Model, which represents a set of classes that represent different parts of an XML document, [11].

The xmlTextWriter class is used to write XML files.

Some of the methods used are given by:

- WriteStartDocument - writes the string <?xml version="1.0">;
- WriteStartElement - writes the tag to open the element;
- WriteEndElement - writes the tag to close the element;
- WriteAttributeString - writes the attribute for the element;
- WriteString - writes the data for the element as a string;
- WriteElementString - writes an element and its string data;
- WriteComment - writes a comment;
- WriteCData- writes the CData part;
- Flush- empty the contents of the printer; and
- Close - closes the xmlTextWriter.

The properties of this class are shown by:

- Formatting- sets to how the file should be formatted, and can be Formatting.Intended or Formatting.None, which is the default;
- Indentation- gives how much each sub-element is indented; and
- WriteElementString writes the start tag, data, and end tag, and automatically writes the trailing element.

The xmlTextWriter and xmlTextReader classes behave in the same way as SteamReader and SteamWriter. The listing below, Listing 1, shows how to create an XML file as a structured formatted database with the members of the Alumni Club. By selecting the "Create Alumni XML file" button, in the Alumni application the alumni.xml file will be created.

The xmlTextReader class is used to read the alumni XML file. The listing further shown (Listing 2) presents how to read the file with the members of the Alumni Club, and it is a fragment of complete listing.

```
xmlTextWriter xmlCreate=new xmlTextWriter(Server.MapPath(alumni.xml), null);
xmlCreate.Formatting=Formatting.Indented;
xmlCreate.WriteStartElement("Alumni");
xmlCreate.WriteComment("This file is created by using xmltext writer class");
xmlCreate.WriteStartElement("AlumniMember");
xmlCreate.WriteAttributeString("ID", "1");
xmlCreate.WriteAttributeString("Name", "Nemad");
xmlCreate.WriteAttributeString("LastName", "Nemadovic");
xmlCreate.WriteAttributeString("Sex", "Male");
xmlCreate.WriteAttributeString("Place", "Zrenjanin");
xmlCreate.WriteAttributeString("Date", "2/1/2019");
xmlCreate.WriteAttributeString("Email", "nnenadovic@gmail.com");
xmlCreate.WriteAttributeString("Profession", "Administrator");
xmlCreate.WriteAttributeString("Employed", "Yes");
xmlCreate.WriteAttributeString("Attainment", "Master of Science");
xmlCreate.WriteEndElement();
xmlCreate.WriteEndElement();
xmlCreate.WriteEndDocument();
xmlCreate.Close();
```

Listing 1 - Creation of XML file with Alumni members.

```
xmlTextReader xmlReader=new(Server.MapPath(alumni.xml), null);
while(xmlReader.Read())
if(xmlReader.NodeType == xmlNodeType.Element)
{
    Response.Write("<b>Element:</b>" + xmlReader.Name);
```

Listing 2 - Reading of XML file with Alumni members.

The properties of the xmlTextReader class are:

- Name - gives a value to the current node;
- Depth - shows the depth in the XML tree;
- Value - gives the value of the current node;
- Item - gives the value of the Attribute;
- EDF - if we are at the end of the file the value is True; and
- NodeType - shows the current node type.

The node types in the xmlNodeType enumeration are:

- Element - represents the initial tag of an XML element;
- EndElement - represents the closing tag of an XML element;
- CDATA - part of the XML node;
- Text - displays text data in an XML node;
- WhiteSpace - white sign between nodes; and
- xmlDeclaration - XML header.

3. THE SECOND APPROACH TO CREATING AN ALUMNI XML FILE

The other approach includes the creation of an application developed in C# programming language. A form for inserting new members of the Alumni Club is presented in Figure 1.

When the Add Alumni Member button is selected, a new xmlDocument is created. That document is written to the file specified by the private PATH field. It is then tested whether the file already exists using the Exists method of the System.IO.File class. If the file does not exist, then a new file is created and the first record is added.

The code below creates the necessary nodes that are added to the Xml document. First, an Xml Declaration is created using the xmlDeclaration class and the Createxml-Declaration method. This method uses three parameters; version, encoding, and whether the file is standalone.

It then creates a comment using the xmlComment class and the CreateComment method and passes the text to the method used in the comment. After that, the root element is created using the CreateElement method and the xmlElement class. The CreateElement method is used to create the root, parent, and child elements. This method takes a single argument as a string representing the name of the given element.

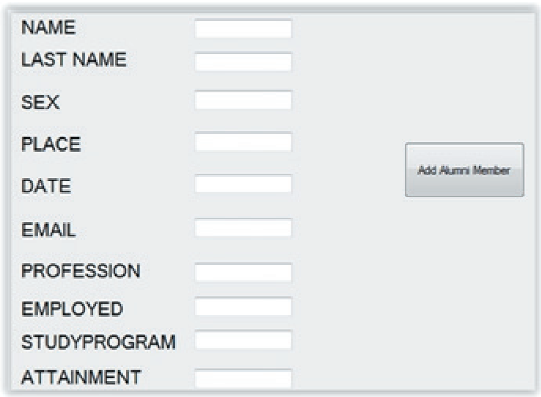

Figure 1 - A form for inserting new Alumni member.

An Alumni element is created and stores the data entered by the user. The Alumni element has one name attribute, while the other elements are children. The CreateAttribute method accepts one argument representing the name of the attribute. Attribute details are stored in the xmlAttribute object.

Using the AppendChild method adds a declaration to the xmlDocument and adds it as the last child element to the document. Then a comment is added immediately below the declaration. Then it starts with the root element and the child elements.

Reading content from XML files is a necessary technique that is used a lot in the case of saving configuration settings that are usually placed in an XML file and then used when loading an application.

The special query language XPath is used to select nodes in an XML document. Using this language, it is not necessary to search the entire tree of XML nodes. The two methods used to select nodes in the XPath programming language are xmlNode.SelectNodes() and xmlNode.SelectSingleNode().

The SelectNodes() method displays an xmlNodeList containing all nodes that match the XPath string.

A part of the code is presented in the following listing – Listing 3.

```
private XmlDocument doc;
private const string Path = @"D:\alumni.xml
private void btnAddAlumniMember_Click(object sender,EventArgs e)
{
    //Create XML document
    doc = new XmlDocument();
    if (!System.IO.File.Exists(PATH))
    {
        XmlDeclaration declaration = doc.CreateXmlDeclaration("1.0","UTF-8", "yes");
        XmlElement root = doc.CreateElement("Alumni");
        XmlElement alumnimember = doc.CreateElement("AlumniMember");
        XmlAttribute name = doc.CreateAttribute("Name");
        XmlElement lastname = doc.CreateElement("LastName");
        XmlElement sex = doc.CreateElement("Sex");
        XmlElement place = doc.CreateElement("Place");
        XmlElement date = doc.CreateElement("Date");
        XmlElement email = doc.CreateElement("Email");
        // Add values to each node
        name.InnerText = txtName.Text;
        lastname.InnerText = txtLastName.Text;
        sex.InnerText = txtSex.Text;
        place.InnerText = txtPlace.Text;
        date.InnerText = txtDate.Text;
        email.InnerText = txtEmail.Text;
        profession.InnerText = txtProfession.Text;
        employed.InnerText = txtEmployed.Text;
        studyprogram.InnerText = txtStudyProgram.Text;
        attainment.InnerText = txtAttainment.Text;
        // Create document
        doc.AppendChild(declaration);
        doc.AppendChild(comment);
        doc.AppendChild(root);
        root.AppendChild(alumnimember);
        student.Attributes.Append(name);
        student.AppendChild(lastname);
        student.AppendChild(sex);
        student.AppendChild(place);
        student.AppendChild(date);
        student.AppendChild(email);
        student.AppendChild(profession);
        student.AppendChild(employed);
        student.AppendChild(studyprogram);
        student.AppendChild(attainment);
        doc.Save(PATH);
      }
}
```

Listing 3 - Creation of XML file with Alumni members from application in C#.

4. CONCLUSION

Taking into account that the number of members of the Computer-Mechanical Engineering Section is constantly growing, it was necessary to create a database for easier data manipulation. The database was created using a SQL server; advanced queries were used, as well as stored procedures, suitable for obtaining various statistical data and other relevant indicators.

This paper shows the creation of an XML file in two ways, by directly selecting an appropriate button, and through the application developed in C#, by entering desired data into text boxes. The chosen XML format enables even more successful data processing due to its use in all programming languages and platforms, as an opportunity to create a stable Web application. For differently formatted XML files, text files, and other formats, it is used XSL language. This language is primarily used to transform XML files into HTML, where the transformation process takes place automatically.

5. REFERENCES

[1] Belgrade Polytechnic [Internet] Available at: https://www.atssb.edu.rs/alumni-bp-o-nama// (Accessed: 10.04.2023).

[2]. Computer Mechanical Engineering Section, [Internet] Available at:
https://www.visokatehnicka.edu.rs/alumni/ (Accessed: 28.04.2023).

[3] Dihovicni Dj., “Creating the Application for the Alumni Project”, in 6th *Scientific-Professional Conference Politehnika*, 10. 12. 2021., pp. 516–520, Belgrade, Serbia.

[4] Dihovicni Dj., Kreculj D.., “Implementing Stored Procedures in the Alumni Project”, in *6th Scientific-Professional Conference Politehnika*, 10. 12. 2021., pp. 521–526, Belgrade, Serbia.

[5] Kumar R., Gupta N., Maharwal H., Charu S., Yadav K., “Critical Analysis of Database Management Using New SQL”, in I*nternational Journal of Computer Science and Mobile Computing*, Vol. 3, pp. 434-438, 2014.

[6] Dihovicni Dj., Medenica M.,“ Fuzzy support model for long pipelines by using DB2 approach”, in *Applied Engineering Letters*, Vol. 2, No. 2, pp. 76-83, 2016.

[7] Dihovicni Dj., Medenica M.., “Development of software package named Step method for robust time delay systems”, in I*nternational Journal of Latest Research in Engineering and Technology*, Vol. 3, No. 8, pp. 36-43, 2017.

[8] Neeraj N.,“Mastering Apache Cassandra”, Pact Publishing Ltd., UK, 2013.

[9] Dihovicni Dj., Medenica M., "Database linear of scalability and high availability while maintaining a system performance", in *10th International Scientific Conference Science and Higher Education in Function of Sustainable Development*, Section 2., pp. 45–53, Uzice, Serbia, 2017.

[10] Le Raman, Baghla S., Monga H., “Method & Implementation of Data Flow”, in *Applied Engineering Letters,* Vol. 1, No. 4, pp. 105-110, 2016.

[11] Dihovični Dj., "Object oriented programming", Technical College, Belgrade, 2015.