

# LAB - SOFTWARE ARCHITECTURE

## Mục lục

<b>1</b>	<b>Cloud hostings</b>	<b>2</b>
1.1	GearHost . . . . .	2
1.1.1	Database . . . . .	2
1.1.2	CloudSite . . . . .	2
1.2	Google Firebase . . . . .	3
<b>2</b>	<b>Architectural styles</b>	<b>3</b>
2.1	nLayers/nTiers architecture . . . . .	3
2.1.1	ADO.NET . . . . .	3
2.1.2	LINQ . . . . .	8
2.2	Client-Server architecture . . . . .	10
2.2.1	Webform . . . . .	10
2.2.2	Remoting . . . . .	13
2.3	Service-Oriented architecture . . . . .	16
2.3.1	Windows Service . . . . .	16
2.3.2	SOAP Service . . . . .	17
2.3.3	RESTful Service . . . . .	31
2.3.4	Backend as a Service . . . . .	39
2.3.5	Microservice . . . . .	46
2.4	Others architecture . . . . .	49
2.4.1	MVC . . . . .	49
2.4.2	Parallel . . . . .	49

# 1 Cloud hostings

## 1.1 GearHost

- Sign up account from GearHost: <https://www.gearhost.com>

### 1.1.1 Database

- Menu Databases » Create Database:

- Database Name: sonkk
- Database Type: MSSQL

» Create Empty Database » Database informations:

- Database Name: sonkk
- Database Server: den1.mssql7.gear.host
- Username: sonkk
- Password: pwd

- SQL Scripting:

```

1 --Create table
2 IF OBJECT_ID('Book') IS NOT NULL DROP TABLE Book;
3 CREATE TABLE Book(
4     Code int PRIMARY KEY IDENTITY ,
5     Name nvarchar(50) NOT NULL ,
6     Author nvarchar(50) NOT NULL ,
7     Price int
8 );
9 --Insert data
10 INSERT INTO Book(Name , Author , Price) VALUES('SA' , 'HSU1' , 111);
11 INSERT INTO Book(Name , Author , Price) VALUES('OOP' , 'HSU2' , 222);
12 INSERT INTO Book(Name , Author , Price) VALUES('OOD' , 'HSU3' , 333);
13 --Select data
14 SELECT * FROM Book;
```

### 1.1.2 CloudSite

- Menu CloudSites » Add CloudSite:

- Application Name: sonkk (<http://sonkk.gear.host>)
- Application Type: Free

» Create CloudSite » Access CloudSite » Publish Tab » Application Publishing Files » Download Visual Studio publishing profile (sonkk.publishsettings file)

## 1.2 Google Firebase

- Sign in google account from Firebase: <http://firebase.google.com> » Go to console » Add project

- Project name: sonkk
- Uncheck on Enable Google Analytics for this project
- Menu Realtime Database » Create Database » Security rules: Start in test mode
- URL: <https://sonkk.firebaseio.com>

## 2 Architectural styles

### 2.1 nLayers/nTiers architecture

#### 2.1.1 ADO.NET

- Create a new project:

- Project type: Windows Forms App (.NET Framework)
- Project name: BookClient
- Solution name: nLayers

#### BookClient project

- Update App.config

```

1 <configuration>
2   ...
3   <connectionStrings>
4     <add name="strCon" providerName="System.Data.SqlClient" connectionString="
5       SERVER=den1.mssql7.gear.host;DATABASE=sonkk;USER=sonkk;PASSWORD=pwd" />
6   </connectionStrings>
7 </configuration>
```

- Create Book.cs (DTO class)

```

1 ...
2 class Book {
3   public int Code { get; set; }
4   public String Name { get; set; }
5   public String Author { get; set; }
6   public int Price { get; set; }
7 }
```

- Create BookForm.cs (presentation class)

**BOOK**

**Books**

Keyword

	Code	Name	Author	Price
▶	1	SA	HSU1	111
	2	OOP	HSU2	222
	3	OOAD	HSU3	333

**Controls**

Book Code	<input type="text" value="1"/>	<input type="button" value="Add"/>
Book Name	<input type="text" value="SA"/>	<input type="button" value="Update"/>
Book Author	<input type="text" value="HSU1"/>	<input type="button" value="Delete"/>
Book Price	<input type="text" value="111"/>	

```

1 ...
2 public partial class BookForm : Form {
3     public BookForm() {
4         InitializeComponent();
5     }
6     private void BookForm_Load(object sender, EventArgs e) {
7         List<Book> books = new BookBUS().GetAll();
8         gridBook.DataSource = books;
9     }
10    private void BookForm_FormClosing(object sender, FormClosingEventArgs e) {
11        Application.Exit();
12    }
13    private void gridBook_SelectionChanged(object sender, EventArgs e) {
14        if (gridBook.SelectedRows.Count > 0) {
15            int code = int.Parse(gridBook.SelectedRows[0].Cells["Code"].Value.ToString());
16            Book book = new BookBUS().GetDetails(code);
17            if (book != null) {
18                txtCode.Text = book.Code.ToString();
19                txtName.Text = book.Name;
20                txtAuthor.Text = book.Author;
21                txtPrice.Text = book.Price.ToString();
22            }
23        }
}

```

```

24 }
25 private void btnSearch_Click(object sender, EventArgs e) {
26     String keyword = txtKeyword.Text.Trim();
27     List<Book> books = new BookBUS().Search(keyword);
28     gridBook.DataSource = books;
29 }
30 private void btnAdd_Click(object sender, EventArgs e) {
31     Book newBook = new Book() {
32         Code = 0,
33         Name = txtName.Text.Trim(),
34         Author = txtAuthor.Text.Trim(),
35         Price = int.Parse(txtPrice.Text.Trim())
36     };
37     bool result = new BookBUS().AddNew(newBook);
38     if (result) {
39         List<Book> books = new BookBUS().GetAll();
40         gridBook.DataSource = books;
41     } else {
42         MessageBox.Show("SORRY BABY!");
43     }
44 }
45 private void btnUpdate_Click(object sender, EventArgs e) {
46     Book newBook = new Book() {
47         Code = int.Parse(txtCode.Text.Trim()),
48         Name = txtName.Text.Trim(),
49         Author = txtAuthor.Text.Trim(),
50         Price = int.Parse(txtPrice.Text.Trim())
51     };
52     bool result = new BookBUS().Update(newBook);
53     if (result) {
54         List<Book> books = new BookBUS().GetAll();
55         gridBook.DataSource = books;
56     } else {
57         MessageBox.Show("SORRY BABY!");
58     }
59 }
60 private void btnDelete_Click(object sender, EventArgs e) {
61     DialogResult dialogResult = MessageBox.Show("ARE YOU SURE ?", "CONFIRMATION"
62         , MessageBoxButtons.YesNo);
63     if (dialogResult == DialogResult.Yes) {
64         int code = int.Parse(txtCode.Text);
65         bool result = new BookBUS().Delete(code);
66         if (result) {
67             List<Book> books = new BookBUS().GetAll();
68             gridBook.DataSource = books;
69         } else {
70             MessageBox.Show("SORRY BABY!");
71         }
72     }
73 }

```

- Create BookBUS.cs (business-logic class)

```

1 ...
2 class BookBUS {
3     public List<Book> GetAll() {
4         List<Book> books = new BookDAO().SelectAll();
5         return books;
6     }
7     public Book GetDetails(int code) {
8         Book book = new BookDAO().SelectByCode(code);
9         return book;
10    }
11    public List<Book> Search(String keyword) {
12        List<Book> books = new BookDAO().SelectByKeyword(keyword);
13        return books;
14    }
15    public bool AddNew(Book newBook) {
16        bool result = new BookDAO().Insert(newBook);
17        return result;
18    }
19    public bool Update(Book newBook) {
20        bool result = new BookDAO().Update(newBook);
21        return result;
22    }
23    public bool Delete(int code) {
24        bool result = new BookDAO().Delete(code);
25        return result;
26    }
27 }
```

- Create BookDAO.cs (data-access class)

```

1 ...
2 using System.Configuration; // add reference System.Configuration.dll
3 using System.Data.SqlClient;
4
5 class BookDAO {
6     public List<Book> SelectAll() {
7         List<Book> books = new List<Book>();
8         String strCon = ConfigurationManager.ConnectionStrings["strCon"].
9             ConnectionString;
10        SqlConnection con = new SqlConnection(strCon);
11        con.Open();
12        String strCom = "SELECT * FROM Book";
13        SqlCommand com = new SqlCommand(strCom, con);
14        SqlDataReader dr = com.ExecuteReader();
15        while (dr.Read()) {
16            Book book = new Book() {
17                Code = (int)dr["Code"],
18                Name = (String)dr["Name"],
19                Author = (String)dr["Author"],
```

```

19     Price = (int)dr["Price"]
20 };
21     books.Add(book);
22 }
23     return books;
24 }
25 public Book SelectByCode(int code) {
26     String strCon = ConfigurationManager.ConnectionStrings["strCon"].
27         ConnectionString;
28     SqlConnection con = new SqlConnection(strCon);
29     con.Open();
30     String strCom = "SELECT * FROM Book WHERE Code=@Code";
31     SqlCommand com = new SqlCommand(strCom, con);
32     com.Parameters.Add(new SqlParameter("@Code", code));
33     SqlDataReader dr = com.ExecuteReader();
34     if (dr.Read()) {
35         Book book = new Book() {
36             Code = (int)dr["Code"],
37             Name = (String)dr["Name"],
38             Author = (String)dr["Author"],
39             Price = (int)dr["Price"]
40         };
41         return book;
42     }
43     return null;
44 }
45 public List<Book> SelectByKeyword(String keyword) {
46     List<Book> books = new List<Book>();
47     String strCon = ConfigurationManager.ConnectionStrings["strCon"].
48         ConnectionString;
49     SqlConnection con = new SqlConnection(strCon);
50     con.Open();
51     String strCom = "SELECT * FROM Book WHERE Name LIKE @Keyword";
52     SqlCommand com = new SqlCommand(strCom, con);
53     com.Parameters.Add(new SqlParameter("@Keyword", "%" + keyword + "%"));
54     SqlDataReader dr = com.ExecuteReader();
55     while (dr.Read()) {
56         Book book = new Book() {
57             Code = (int)dr["Code"],
58             Name = (String)dr["Name"],
59             Author = (String)dr["Author"],
60             Price = (int)dr["Price"]
61         };
62         books.Add(book);
63     }
64     return books;
65 }
66 public bool Insert(Book newBook) {
67     String strCon = ConfigurationManager.ConnectionStrings["strCon"].
68         ConnectionString;
69     SqlConnection con = new SqlConnection(strCon);

```

```

67     con.Open();
68     String strCom = "INSERT INTO Book VALUES(@Name ,@Author ,@Price)";
69     SqlCommand com = new SqlCommand(strCom, con);
70     com.Parameters.Add(new SqlParameter("@Name", newBook.Name));
71     com.Parameters.Add(new SqlParameter("@Author", newBook.Author));
72     com.Parameters.Add(new SqlParameter("@Price", newBook.Price));
73     try { return com.ExecuteNonQuery() > 0; }
74     catch { return false; }
75 }
76 public bool Update(Book newBook) {
77     String strCon = ConfigurationManager.ConnectionStrings["strCon"].
78         ConnectionString;
79     SqlConnection con = new SqlConnection(strCon);
80     con.Open();
81     String strCom = "UPDATE Book SET Name=@Name ,Author=@Author ,Price=@Price
82         WHERE Code=@Code";
83     SqlCommand com = new SqlCommand(strCom, con);
84     com.Parameters.Add(new SqlParameter("@Name", newBook.Name));
85     com.Parameters.Add(new SqlParameter("@Author", newBook.Author));
86     com.Parameters.Add(new SqlParameter("@Price", newBook.Price));
87     com.Parameters.Add(new SqlParameter("@Code", newBook.Code));
88     try { return com.ExecuteNonQuery() > 0; }
89     catch { return false; }
90 }
91 public bool Delete(int code) {
92     String strCon = ConfigurationManager.ConnectionStrings["strCon"].
93         ConnectionString;
94     SqlConnection con = new SqlConnection(strCon);
95     con.Open();
96     String strCom = "DELETE FROM Book WHERE Code=@Code";
97     SqlCommand com = new SqlCommand(strCom, con);
98     com.Parameters.Add(new SqlParameter("@Code", code));
99     try { return com.ExecuteNonQuery() > 0; }
100    catch { return false; }
101 }

```

## 2.1.2 LINQ

- Menu Tools » Get Tools and Features... » Individual components tab » Search by "LINQ to SQL tools" » Install
- Create a new project:
  - Project type: Windows Forms App (.NET Framework)
  - Project name: BookClient
  - Solution name: LINQ

### BookClient project

- Create MyDB.dbml (LINQ to SQL Classes)

- Add Data Connection:
    - Server Explorer tab » Right-click Data Connections » Add Connection... » Data source: Microsoft SQL Server
    - Server name: den1.mssql7.gear.host
    - Authentication: SQL Server Authentication
    - Username: sonkk & Password: pwd & Check on Save my password
    - Select or enter a database name: sonkk
  - Drag and drop Book table in DB server into MyDB.dbml designer
  - Warning "The connection string ... a clear text password ... security" » No
  - Open MyDB.designer.cs file » Review MyDBDataContext class & Book class (DTO class)

```
1 ...
2 using System.Configuration; // add reference System.Configuration.dll
3
4 class BookDAO {
5     private MyDBDataContext db = new MyDBDataContext(ConfigurationManager.
6         ConnectionStrings["strCon"].ConnectionString);
7
8     public List<Book> SelectAll() {
9         db.ObjectTrackingEnabled = false; // fix error "System.Runtime.Serialization
10            .SerializationException: Type 'System.Data.Linq.ChangeTracker+
11            StandardChangeTracker' in Assembly '....' is not marked as serializable"
12         List<Book> books = db.Books.ToList();
13         return books;
14     }
15
16     public Book SelectByCode(int code) {
17         Book book = db.Books.SingleOrDefault(b => b.Code == code);
18         return book;
19     }
20
21     public List<Book> SelectByKeyword(String keyword) {
22         List<Book> books = db.Books.Where(b => b.Name.Contains(keyword)).ToList();
23         return books;
24     }
25
26     public bool Insert(Book newBook) {
27         try {
28             db.Books.InsertOnSubmit(newBook);
29             db.SubmitChanges();
30             return true;
31         }
32         catch { return false; }
33     }
34
35     public bool Update(Book newBook) {
36         Book dbBook = db.Books.SingleOrDefault(b => b.Code == newBook.Code);
37         dbBook.Name = newBook.Name;
38         db.SubmitChanges();
39         return true;
40     }
41 }
```

```

30     if (dbBook != null) {
31         try {
32             dbBook.Name = newBook.Name;
33             dbBook.Author = newBook.Author;
34             dbBook.Price = newBook.Price;
35             db.SubmitChanges();
36             return true;
37         }
38         catch { return false; }
39     }
40     return false;
41 }
42 public bool Delete(int code) {
43     Book dbBook = db.Books.SingleOrDefault(b => b.Code == code);
44     if (dbBook != null) {
45         try {
46             db.Books.DeleteOnSubmit(dbBook);
47             db.SubmitChanges();
48             return true;
49         }
50         catch { return false; }
51     }
52     return false;
53 }
54 }
```

## 2.2 Client-Server architecture

### 2.2.1 Webform

- Create a new project:

- Project type: ASP.NET Web Application (.NET Framework)
- Project name: BookSite
- Solution name: Webform
- Empty project

#### BookSite project

- Update Web.config

```

1 <configuration>
2 ...
3 <connectionStrings>
4     <add name="strCon" providerName="System.Data.SqlClient" connectionString="
5         SERVER=den1.mssql17.gear.host;DATABASE=sonkk;USER=sonkk;PASSWORD=pwd" />
6 </connectionStrings>
</configuration>
```

- Create BookForm.aspx (presentation page)

```

1 ...
2 <form id="form1" runat="server">
3   <h1>BOOK LIST</h1>
4   <asp:TextBox ID="txtKeyword" runat="server"></asp:TextBox>
5   <asp:Button ID="btnSearch" runat="server" Text="SEARCH" OnClick="
6     btnSearch_Click" /><br/><br/>
7   <asp:GridView ID="gvBooks" runat="server" AutoGenerateSelectButton="True"
8     OnSelectedIndexChanged="gvBooks_SelectedIndexChanged"></asp:GridView><br/>
9   <h1>BOOK DETAILS</h1>
10  <table>
11    <tr>
12      <td>Code</td>
13      <td><asp:TextBox ID="txtCode" runat="server" ReadOnly="true"></asp:TextBox>
14        ></td>
15    </tr>
16    <tr>
17      <td>Name</td>
18      <td><asp:TextBox ID="txtName" runat="server"></asp:TextBox></td>
19    </tr>
20    <tr>
21      <td>Author</td>
22      <td><asp:TextBox ID="txtAuthor" runat="server"></asp:TextBox></td>
23    </tr>
24    <tr>
25      <td>Price</td>
26      <td><asp:TextBox ID="txtPrice" runat="server"></asp:TextBox></td>
27    </tr>
28  </table><br/>
29  <asp:Button ID="btnAdd" runat="server" Text="ADD NEW" OnClick="btnAdd_Click" /
30  >
31  <asp:Button ID="btnUpdate" runat="server" Text="UPDATE" OnClick="
32    btnUpdate_Click" />
33  <asp:Button ID="btnDelete" runat="server" Text="DELETE" OnClick="
34    btnDelete_Click" OnClientClick="return confirm('ARE YOU SURE?');" />
35 </form>
36 ...

```

```

1 ...
2 public partial class Default : System.Web.UI.Page {
3   protected void Page_Load(object sender, EventArgs e) {
4     if (!Page.IsPostBack) {
5       List<Book> books = new BookBUS().GetAll();
6       gvBooks.DataSource = books;
7       gvBooks.DataBind();
8     }
9   }
10  protected void gvBooks_SelectedIndexChanged(object sender, EventArgs e) {
11    txtCode.Text = gvBooks.SelectedRow.Cells[1].Text.Trim();
12    txtName.Text = gvBooks.SelectedRow.Cells[2].Text.Trim();
13    txtAuthor.Text = gvBooks.SelectedRow.Cells[3].Text.Trim();

```

```

14     txtPrice.Text = gvBooks.SelectedRow.Cells[4].Text.Trim();
15 }
16 protected void btnSearch_Click(object sender, EventArgs e) {
17     String keyword = txtKeyword.Text.Trim();
18     List<Book> books = new BookBUS().Search(keyword);
19     gvBooks.DataSource = books;
20     gvBooks.DataBind();
21 }
22 protected void btnAdd_Click(object sender, EventArgs e) {
23     Book newBook = new Book() {
24         Code = int.Parse(txtCode.Text.Trim()),
25         Name = txtName.Text.Trim(),
26         Author = txtAuthor.Text.Trim(),
27         Price = int.Parse(txtPrice.Text.Trim())
28     };
29     bool result = new BookBUS().AddNew(newBook);
30     if (result) {
31         List<Book> books = new BookBUS().GetAll();
32         gvBooks.DataSource = books;
33         gvBooks.DataBind();
34     }
35 }
36 protected void btnUpdate_Click(object sender, EventArgs e) {
37     Book newBook = new Book() {
38         Code = int.Parse(txtCode.Text.Trim()),
39         Name = txtName.Text.Trim(),
40         Author = txtAuthor.Text.Trim(),
41         Price = int.Parse(txtPrice.Text.Trim())
42     };
43     bool result = new BookBUS().Update(newBook);
44     if (result) {
45         List<Book> books = new BookBUS().GetAll();
46         gvBooks.DataSource = books;
47         gvBooks.DataBind();
48     }
49 }
50 protected void btnDelete_Click(object sender, EventArgs e) {
51     int code = int.Parse(txtCode.Text.Trim());
52     bool result = new BookBUS().Delete(code);
53     if (result) {
54         List<Book> books = new BookBUS().GetAll();
55         gvBooks.DataSource = books;
56         gvBooks.DataBind();
57     }
58 }
59 }
```

- Deploy on GearHost: Right-click Project » Publish...

- Target: Import Profile

- Browse to sonkk.publishsettings file
- Edit » Settings » File Publish Options » Check on Remove additional files at destination
- Publish » http://sonkk.gear.host/BookForm.aspx

### 2.2.2 Remoting

- Create 3 new projects:

- Shared:
  - Project type: Class Library (.NET Framework)
  - Project name: BookShared
  - Solution name: Remoting
- Server:
  - Project type: Console App (.NET Framework)
  - Project name: BookServer
  - Solution name: Remoting
- Client:
  - Project type: Windows Forms App (.NET Framework)
  - Project name: BookClient
  - Solution name: Remoting
- Add Reference BookShared into BookServer and BookClient

#### BookShared project

- Create Book.cs (DTO class)

```

1 ...
2 [Serializable]
3 public class Book {
4     public int Code { get; set; }
5     public String Name { get; set; }
6     public String Author { get; set; }
7     public int Price { get; set; }
8 }
```

OR create MyDB.dbml (LINQ to SQL Classes) » Update MyDB.designer.cs

```

1 ...
2 [Serializable]
3 [global::System.Data.Linq.Mapping.TableAttribute(Name="dbo.Book")]
4 public partial class Book : INotifyPropertyChanging, INotifyPropertyChanged {
5     ...
6 }
7 ...
```

- Create IBookBUS.cs (interface)

```

1 ...
2 public interface IBookBUS {
3     List<Book> GetAll();
4     Book GetDetails(int code);
5     List<Book> Search(String keyword);
6     bool AddNew(Book newBook);
7     bool Update(Book newBook);
8     bool Delete(int code);
9 }
```

## BookServer project

- Update Program.cs (entry-point)

```

1 ...
2 using System.Runtime.Remoting; // add reference System.Runtime.Remoting.dll
3 using System.Runtime.Remoting.Channels;
4 using System.Runtime.Remoting.Channels.Tcp;
5
6 class Program {
7     static void Main(string[] args) {
8         ChannelServices.RegisterChannel(new TcpChannel(6969), false);
9         RemotingConfiguration.RegisterWellKnownServiceType(typeof(BookBUS), "xxx",
10             WellKnownObjectMode.SingleCall);
11         RemotingConfiguration.CustomErrorsMode = CustomErrorsModes.Off;
12         Console.WriteLine("SERVER is started ...");
13         Console.Read();
14     }
}
```

- Create BookBUS.cs (business-logic class)

```

1 ...
2 using Remoting.BookShared;
3 class BookBUS : MarshalByRefObject, IBookBUS {
4     ...
5 }
```

- Create BookDAO.cs (data-access class)

## BookClient project

- Create BookForm.cs (presentation class)

```

1 ...
2 using Remoting.BookShared;
3
4 public partial class BookForm : Form {
5     const String uri = "tcp://ip_server:6969/xxx";
6     IBookBUS bookBUS = (IBookBUS)Activator.GetObject(typeof(IBookBUS), uri);
```

```

7
8     private void BookForm_Load(object sender, EventArgs e) {
9         List<Book> books = bookBUS.GetAll();
10        ...
11    }
12    private void gridBook_SelectionChanged(object sender, EventArgs e) {
13        ...
14        Book book = bookBUS.GetDetails(code);
15        ...
16    }
17    private void btnSearch_Click(object sender, EventArgs e) {
18        ...
19        List<Book> books = bookBUS.Search(keyword);
20        ...
21    }
22    private void btnAdd_Click(object sender, EventArgs e) {
23        ...
24        bool result = bookBUS.AddNew(newBook);
25        if (result) {
26            List<Book> books = bookBUS.GetAll();
27            ...
28        }
29        ...
30    }
31    private void btnUpdate_Click(object sender, EventArgs e) {
32        ...
33        bool result = bookBUS.Update(newBook);
34        if (result) {
35            List<Book> books = bookBUS.GetAll();
36            ...
37        }
38        ...
39    }
40    private void btnDelete_Click(object sender, EventArgs e) {
41        ...
42        bool result = bookBUS.Delete(code);
43        if (result) {
44            List<Book> books = bookBUS.GetAll();
45            ...
46        }
47        ...
48    }
49 }
```

- Deploy on 2 separate machines:

- Server: start BookServer
- Client: start BookClient

## 2.3 Service-Oriented architecture

### 2.3.1 Windows Service

- Create 3 new projects:

- Shared:
  - Project type: Class Library (.NET Framework)
  - Project name: BookShared
  - Solution name: WindowsService
- Service:
  - Project type: Windows Service (.NET Framework)
  - Project name: BookService
  - Solution name: WindowsService
- Client:
  - Project type: Windows Forms App (.NET Framework)
  - Project name: BookClient
  - Solution name: WindowsService
- Add Reference BookShared into BookService and BookClient

#### BookService project

- Update MainService.cs (entry-point)

```

1 ...
2 using System.Runtime.Remoting; // add reference System.Runtime.Remoting.dll
3 using System.Runtime.Remoting.Channels;
4 using System.Runtime.Remoting.Channels.Tcp;
5
6 public partial class MainService : ServiceBase {
7 ...
8     protected override void OnStart(string[] args) {
9         ChannelServices.RegisterChannel(new TcpChannel(6969), false);
10        RemotingConfiguration.RegisterWellKnownServiceType(typeof(BookBUS), "xxx",
11            WellKnownObjectMode.SingleCall);
12        RemotingConfiguration.CustomErrorsMode = CustomErrorsModes.Off;
13    }
14 }
```

- Open MainService.cs [Design] » Right-click » Add Installer

- serviceInstaller1:
  - ServiceName: BookService
  - StartType: Automatic
- serviceProcessInstaller1:

- Account: LocalSystem

- Install BookService: open Command Prompt (Run as administrator)

```
1 cd C:\Windows\Microsoft.NET\Framework64\v4.0.30319
2 InstallUtil /i $PROJECT_PATH$/bin/Debug/BookService.exe
```

- Uninstall BookService: open Command Prompt (Run as administrator)

```
1 cd C:\Windows\Microsoft.NET\Framework64\v4.0.30319
2 InstallUtil /u $PROJECT_PATH$/bin/Debug/BookService.exe
```

- Deploy on 2 separate machines:

- Server: start BookService in Windows services (OR restart Server)
- Client: start BookClient

### 2.3.2 SOAP Service

- Create 4 new projects:

- Service:
  - Project type: ASP.NET Web Application (.NET Framework)
  - Project name: BookSite
  - Solution name: SoapService
  - Empty project
- Winform Client:
  - Project type: Windows Forms App (.NET Framework)
  - Project name: BookClient
  - Solution name: SoapService
- HTML Client:
  - Project type: HTML + Javascript
  - Project name: HtmlClient
  - Solution name: SoapService
- Mobile Client:
  - Project type: React Native
  - Project name: MobileClient
  - Solution name: SoapService

## BookSite project

- Create BookService.asmx (soap-service)

```

1 ...
2 [WebService(Namespace = "http://sonkk.org/")]
3 public class BookService : System.Web.Services.WebService {
4     [WebMethod]
5     public List<Book> GetAll() {
6         List<Book> books = new BookDAO().SelectAll();
7         return books;
8     }
9     [WebMethod]
10    public Book GetDetails(int code) {
11        Book book = new BookDAO().SelectByCode(code);
12        return book;
13    }
14    [WebMethod]
15    public List<Book> Search(String keyword) {
16        List < Book > books = new BookDAO().SelectByKeyword(keyword);
17        return books;
18    }
19    [WebMethod]
20    public bool AddNew(Book newBook) {
21        bool result = new BookDAO().Insert(newBook);
22        return result;
23    }
24    [WebMethod]
25    public bool Update(Book newBook) {
26        bool result = new BookDAO().Update(newBook);
27        return result;
28    }
29    [WebMethod]
30    public bool Delete(int code) {
31        bool result = new BookDAO().Delete(code);
32        return result;
33    }
34 }
```

- Run on localhost: Ctrl+F5 » https://localhost:port/BookService.asmx
- Deploy on GearHost: Right-click Project » Publish... » http://sonkk.gear.host/BookService.asmx

- Postman:

- (POST) http://sonkk.gear.host/BookService.asmx
- Header: Content-Type: text/xml
- Body (raw+XML):

– GetAll

```
1  <?xml version="1.0" encoding="utf-8"?>
```

```

2 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://
4   schemas.xmlsoap.org/soap/envelope/">
5   <soap:Body>
6     < GetAll xmlns="http://sonkk.org/" />
7   </soap:Body>
8 </soap:Envelope>
```

## – GetDetails

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://
4   schemas.xmlsoap.org/soap/envelope/">
5   <soap:Body>
6     <GetDetails xmlns="http://sonkk.org/">
7       <code>1</code>
8     </GetDetails>
9   </soap:Body>
10 </soap:Envelope>
```

## – Search

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://
4   schemas.xmlsoap.org/soap/envelope/">
5   <soap:Body>
6     <Search xmlns="http://sonkk.org/">
7       <keyword>a</keyword>
8     </Search>
9   </soap:Body>
10 </soap:Envelope>
```

## – AddNew

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://
4   schemas.xmlsoap.org/soap/envelope/">
5   <soap:Body>
6     <AddNew xmlns="http://sonkk.org/">
7       <newBook>
8         <Code>4</Code>
9         <Name>MAD</Name>
10        <Author>HSU4</Author>
11        <Price>444</Price>
12      </newBook>
13    </AddNew>
14  </soap:Body>
15 </soap:Envelope>
```

## – Update

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://
   schemas.xmlsoap.org/soap/envelope/">
3   <soap:Body>
4     <Update xmlns="http://sonkk.org/">
5       <newBook>
6         <Code>4</Code>
7         <Name>ADM</Name>
8         <Author>4HSU</Author>
9         <Price>123</Price>
10        </newBook>
11      </Update>
12    </soap:Body>
13 </soap:Envelope>
```

- Delete

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://
   schemas.xmlsoap.org/soap/envelope/">
3   <soap:Body>
4     <Delete xmlns="http://sonkk.org/">
5       <code>4</code>
6     </Delete>
7   </soap:Body>
8 </soap:Envelope>
```

## BookClient project

- Add reference to BookService:

- Add Service Reference » Advanced » Add Web Reference...
- URL: http://sonkk.gear.host/BookService.asmx » Go
- Web reference name: gearhost » Add Reference

- Create BookForm.cs (presentation class)

```

1 ...
2 public partial class BookForm : Form {
3   gearhost.BookService bookService = new gearhost.BookService();
4
5   private void BookForm_Load(object sender, EventArgs e) {
6     List<gearhost.Book> books = bookService.GetAll();
7     ...
8   }
9   private void gridBook_SelectionChanged(object sender, EventArgs e) {
10     ...
```

```

11     gearhost.Book book = bookService.GetDetails(code);
12     ...
13 }
14 private void btnSearch_Click(object sender, EventArgs e) {
15     ...
16     List<gearhost.Book> books = bookService.Search(keyword);
17     ...
18 }
19 private void btnAdd_Click(object sender, EventArgs e) {
20     ...
21     bool result = bookService.AddNew(newBook);
22     if (result) {
23         List<gearhost.Book> books = bookService.GetAll();
24         ...
25     }
26     ...
27 }
28 private void btnUpdate_Click(object sender, EventArgs e) {
29     ...
30     bool result = bookService.Update(newBook);
31     if (result) {
32         List<gearhost.Book> books = bookService.GetAll();
33         ...
34     }
35     ...
36 }
37 private void btnDelete_Click(object sender, EventArgs e) {
38     ...
39     bool result = bookService.Delete(code);
40     if (result) {
41         List<gearhost.Book> books = bookService.GetAll();
42         ...
43     }
44     ...
45 }
46 }
```

## HtmlClient project

- Create books.html (presentation page)

```

1 <html>
2 <head>
3     <script src="https://unpkg.com/axios/dist/axios.min.js"></script>
4     <script src="https://cdn.rawgit.com/abdmob/x2js/master/xml2json.js"></script>
5     <script src="books.js"></script>
6 </head>
7 <body onload="page_Load()">
8     <h2>BOOK LIST</h2>
9     <input type="text" id="txtKeyword" placeholder="Keyword" />&nbsp;
10    <input type="button" value="SEARCH" onclick="btnSearch_Click()" /><br/><br/>
```

```

11 <table id="lstBooks" border="1"></table><br/>
12 <h2>BOOK DETAILS</h2>
13 <table>
14   <tr>
15     <td>Code</td>
16     <td><input type="text" id="txtCode" readonly="true" /></td>
17   </tr>
18   <tr>
19     <td>Name</td>
20     <td><input type="text" id="txtName" /></td>
21   </tr>
22   <tr>
23     <td>Author</td>
24     <td><input type="text" id="txtAuthor" /></td>
25   </tr>
26   <tr>
27     <td>Price</td>
28     <td><input type="text" id="txtPrice" /></td>
29   </tr>
30   <tr>
31     <td colspan="2">
32       <input type="button" value="ADD NEW" onclick="btnAdd_Click()" />
33       <input type="button" value="UPDATE" onclick="btnUpdate_Click()" />
34       <input type="button" value="DELETE" onclick="btnDelete_Click()" />
35     </td>
36   </tr>
37 </table>
38 </body>
39 </html>

```

- Create books.js (event-handlers)

```

1 const CORS = "https://cors-anywhere.herokuapp.com/";
2 const URI = "http://sonkk.gear.host/BookService.asmx";
3 const config = { headers: { 'Content-Type': 'text/xml' } };

4 /* event-handler methods */
5 function page_Load() {
6   getAll();
7 }
8
9 function lnkID_Click(code) {
10   getDetails(code);
11 }
12
13 function btnSearch_Click() {
14   var keyword = document.getElementById("txtKeyword").value.trim();
15   if (keyword.length > 0) {
16     search(keyword);
17   } else {
18     getAll();
19   }
}

```

```

20 function btnAdd_Click() {
21     var newBook = {
22         Code: 0,
23         Name: document.getElementById("txtName").value,
24         Author: document.getElementById("txtAuthor").value,
25         Price: document.getElementById("txtPrice").value
26     };
27     addNew(newBook);
28 }
29 function btnUpdate_Click() {
30     var newBook = {
31         Code: document.getElementById("txtCode").value,
32         Name: document.getElementById("txtName").value,
33         Author: document.getElementById("txtAuthor").value,
34         Price: document.getElementById("txtPrice").value
35     };
36     update(newBook);
37 }
38 function btnDelete_Click() {
39     if (confirm("ARE YOU SURE ?")) {
40         var code = document.getElementById("txtCode").value;
41         delete(code);
42     }
43 }
44 /* fetch SOAP methods */
45 function getAll() {
46     var body = `<?xml version="1.0" encoding="utf-8"?>
47 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
48     http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap
49     /envelope">
50     <soap:Body>
51         <GetAll xmlns="http://sonkk.org/" />
52     </soap:Body>
53 </soap:Envelope>`;
54     axios.post(CORS + URI + "?op=GetAll", body, config).then((response) => {
55         var xmlData = response.data;
56         //alert(xmlData); // for DEBUG
57         var jsonData = new X2JS().xml_str2json(xmlData);
58         //alert(JSON.stringify(jsonData)); // for DEBUG
59         var books = jsonData.Envelope.Body.GetAllResponse.GetAllResult.Book;
60         //alert(JSON.stringify(books)); // for DEBUG
61         renderBookList(books);
62     });
63 }
64 function getDetails(code) {
65     var body = `<?xml version="1.0" encoding="utf-8"?>
66 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
67     http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap
68     /envelope">
69     <soap:Body>
70         <GetDetails xmlns="http://sonkk.org/">

```

```

67     <code>${code}</code>
68   </GetDetails>
69 </soap:Body>
70 </soap:Envelope>`;
71   axios.post(CORS + URI + "?op=GetDetails", body, config).then((response) => {
72     var xmlData = response.data;
73     var jsonData = new X2JS().xml_str2json(xmlData);
74     var book = jsonData.Envelope.Body.GetDetailsResponse.GetDetailsResult;
75     renderBookDetails(book);
76   });
77 }
78 function search(keyword) {
79   var body = `<?xml version="1.0" encoding="utf-8"?>
80 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
81   http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/
82   /envelope/">
83 <soap:Body>
84   <Search xmlns="http://sonkk.org/">
85     <keyword>${keyword}</keyword>
86   </Search>
87 </soap:Body>
88 </soap:Envelope>`;
89   axios.post(CORS + URI + "?op=Search", body, config).then((response) => {
90     var xmlData = response.data;
91     var jsonData = new X2JS().xml_str2json(xmlData);
92     var books = jsonData.Envelope.Body.SearchResponse.SearchResult.Book;
93     renderBookList(books);
94   });
95 }
96 function addNew(newBook) {
97   var body = `<?xml version="1.0" encoding="utf-8"?>
98 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
99   http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/
100  /envelope/">
101 <soap:Body>
102   <AddNew xmlns="http://sonkk.org/">
103     <newBook>
104       <Code>${newBook.Code}</Code>
105       <Name>${newBook.Name}</Name>
106       <Author>${newBook.Author}</Author>
107       <Price>${newBook.Price}</Price>
108     </newBook>
109   </AddNew>
110 </soap:Body>
111 </soap:Envelope>`;
112   axios.post(CORS + URI + "?op=AddNew", body, config).then((response) => {
113     var xmlData = response.data;
114     var jsonData = new X2JS().xml_str2json(xmlData);
115     var result = jsonData.Envelope.Body.AddNewResponse.AddNewResult;
116     if (result) {
117       getAll();
118     }
119   });
120 }
121 
```

```

114     clearTextboxes();
115 } else {
116     alert('SORRY BABY!');
117 }
118 });
119 }
120 function update(newBook) {
121     var body = `<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap
/envelope/">
<soap:Body>
<Update xmlns="http://sonkk.org/">
<newBook>
<Code>${newBook.Code}</Code>
<Name>${newBook.Name}</Name>
<Author>${newBook.Author}</Author>
<Price>${newBook.Price}</Price>
</newBook>
</Update>
</soap:Body>
</soap:Envelope>`;
134     axios.post(CORS + URI + "?op=Update", body, config).then((response) => {
135         var xmlData = response.data;
136         var jsonData = new X2JS().xml_str2json(xmlData);
137         var result = jsonData.Envelope.Body.UpdateResponse.UpdateResult;
138         if (result) {
139             getAll();
140             clearTextboxes();
141         } else {
142             alert('SORRY BABY!');
143         }
144     });
145 }
146 function delete(code) {
147     var body = `<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap
/envelope/">
<soap:Body>
<Delete xmlns="http://sonkk.org/">
<code>${code}</code>
</Delete>
</soap:Body>
</soap:Envelope>`;
155     axios.post(CORS + URI + "?op=Delete", body, config).then((response) => {
156         var xmlData = response.data;
157         var jsonData = new X2JS().xml_str2json(xmlData);
158         var result = jsonData.Envelope.Body.DeleteResponse.DeleteResult;
159         if (result) {
160             getAll();

```

```

161     clearTextboxes();
162 } else {
163     alert('SORRY BABY!');
164 }
165 });
166 }
167 /* helper methods */
168 function renderBookList(books) {
169     var rows = "";
170     for (var book of books) {
171         rows += "<tr onclick='lnkID_Click(" + book.Code + ")' style='cursor:pointer
172             '>";
173         rows += "<td>" + book.Code + "</td>";
174         rows += "<td>" + book.Name + "</td>";
175         rows += "<td>" + book.Author + "</td>";
176         rows += "<td>" + book.Price + "</td>";
177         rows += "</tr>";
178     }
179     var header = "<tr><th>Code</th><th>Name</th><th>Author</th><th>Price</th></tr>
180         ";
181     document.getElementById("lstBooks").innerHTML = header + rows;
182 }
183 function renderBookDetails(book) {
184     document.getElementById("txtCode").value = book.Code;
185     document.getElementById("txtName").value = book.Name;
186     document.getElementById("txtAuthor").value = book.Author;
187     document.getElementById("txtPrice").value = book.Price;
188 }
189 function clearTextboxes() {
190     document.getElementById("txtCode").value = '';
191     document.getElementById("txtName").value = '';
192     document.getElementById("txtAuthor").value = '';
193     document.getElementById("txtPrice").value = '';
194 }
```

## MobileClient project

- Create React Native project by Expo-cli

```

1 expo init MobileClient
2 cd MobileClient
3 npm install axios --save
4 npm install react-native-xml2js --save
5 expo start -c
```

- Update App.js (entry-point)

```

1 import React, { Component } from 'react';
2 import Book from './components/BookComponent';
3
4 class App extends Component {
```

```

5   render() {
6     return (<Book />);
7   }
8 }
9 export default App;

```

- Create components/BookComponent.js

```

1 import React, { Component } from 'react';
2 import { StyleSheet, View, ActivityIndicator, FlatList, TouchableOpacity, Text,
3   TextInput, Button, Alert } from 'react-native';
4 import axios from 'axios';
5 import xml2js from 'react-native-xml2js';
6
7 class Book extends Component {
8   constructor(props) {
9     super(props);
10    this.URI = 'http://sonkk.gear.host/BookService.asmx';
11    this.config = { headers: { 'Content-Type': 'text/xml' } };
12    this.state = { isLoading: true, dataSource: [], txtCode: '', txtName: '',
13      txtAuthor: '', txtPrice: '' };
14  }
15  render() {
16    if (this.state.isLoading) {
17      return (
18        <View style={styles.loading}>
19          <ActivityIndicator />
20        </View>
21      );
22    } else {
23      return (
24        <View style={styles.container}>
25          <View style={styles.bookList}>
26            <Text>BOOK LIST</Text>
27            <FlatList contentContainerStyle={styles.flatList}
28              data={this.state.dataSource}
29              keyExtractor={(item) => item.Code.toString()}
30              renderItem={({ item }) => (
31                <TouchableOpacity onPress={() => this.pressOnRow(item)}>
32                  <Text style={styles.rowItem}>{item.Code} - {item.Name} - {item
33                    .Author} - {item.Price}</Text>
34                  </TouchableOpacity>
35                )} />
36          </View>
37          <View style={styles.line} />
38          <View style={styles.bookDetails}>
39            <Text>BOOK DETAILS</Text>
40            <TextInput style={styles.input} underlineColorAndroid='transparent'
41              placeholder='Code' autoCapitalize='none' editable={false}
42              value={this.state.txtCode} onChangeText={(txtCode) => this.
43                setState({ txtCode })} />

```

```

39         <TextInput style={styles.input} underlineColorAndroid='transparent'
40             placeholder='Name' autoCapitalize='none'
41             value={this.state.txtName} onChangeText={(txtName) => this.
42                 setState({ txtName })} />
43         <TextInput style={styles.input} underlineColorAndroid='transparent'
44             placeholder='Author' autoCapitalize='none'
45             value={this.state.txtAuthor} onChangeText={(txtAuthor) => this.
46                 setState({ txtAuthor })} />
47         <TextInput style={styles.input} underlineColorAndroid='transparent'
48             placeholder='Price' autoCapitalize='none'
49             value={this.state.txtPrice} onChangeText={(txtPrice) => this.
50                 setState({ txtPrice })} />
51     <View style={styles.buttons}>
52         <Button title='ADD NEW' onPress={this.btnAdd} />
53         <Button title='UPDATE' onPress={this.btnUpdate} />
54         <Button title='DELETE' onPress={this.btnDelete} />
55     </View>
56   </View>
57 )
58 }
59 componentDidMount() {
60   this.getAll();
61 }
62 /* event-handler methods */
63 pressOnRow(item) {
64   var code = item.Code;
65   this.getDetails(code);
66 }
67 btnAdd = async () => {
68   var newBook = { Code: 0, Name: this.state.txtName, Author: this.state.
69     txtAuthor, Price: this.state.txtPrice };
70   this.addNew(newBook);
71 }
72 btnUpdate = () => {
73   var newBook = { Code: this.state.txtCode, Name: this.state.txtName, Author:
74     this.state.txtAuthor, Price: this.state.txtPrice };
75   this.update(newBook);
76 }
77 btnDelete = () => {
78   Alert.alert('', 'ARE YOU SURE ?',
79   [
80     { text: 'CANCEL', style: 'cancel' },
81     { text: 'OK', onPress: () => this.delete(this.state.txtCode) }
82   ],
83   { cancelable: false }
84 );
85 }
86 /* fetch SOAP methods */
87 async getAll() {

```

```

82     var body = `<?xml version="1.0" encoding="utf-8"?>
83 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
84   http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/
85   /envelope/">
86   <soap:Body>
87     <GetAll xmlns="http://sonkk.org/" />
88   </soap:Body>
89 </soap:Envelope>`;
90   axios.post(this.URI + '?op=GetAll', body, this.config).then((response) => {
91     var xmlData = response.data;
92     xml2js.parseString(xmlData, { explicitArray: false, ignoreAttrs: true }, (err, result) => {
93       var books = result['soap:Envelope']['soap:Body'].GetAllResponse.
94         GetAllResult.Book;
95       this.setState({ isLoading: false, dataSource: books });
96     });
97   });
98
99   async getDetails(code) {
100     var body = `<?xml version="1.0" encoding="utf-8"?>
101 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
102   http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/
103   /envelope/">
104   <soap:Body>
105     <GetDetails xmlns="http://sonkk.org/">
106       <code>${code}</code>
107     </GetDetails>
108   </soap:Body>
109 </soap:Envelope>`;
110   axios.post(this.URI + '?op=GetDetails', body, this.config).then((response)
111     => {
112       var xmlData = response.data;
113       xml2js.parseString(xmlData, { explicitArray: false, ignoreAttrs: true }, (err, result) => {
114         var book = result['soap:Envelope']['soap:Body'].GetDetailsResponse.
115           GetDetailsResult;
116         this.setState({ txtCode: book.Code.toString(), txtName: book.Name,
117           txtAuthor: book.Author, txtPrice: book.Price.toString() });
118       });
119     });
120   }
121
122   async addNew(newBook) {
123     var body = `<?xml version="1.0" encoding="utf-8"?>
124 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
125   http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/
126   /envelope/">
127   <soap:Body>
128     <AddNew xmlns="http://sonkk.org/">
129       <newBook>
130         <Code>${newBook.Code}</Code>
131         <Name>${newBook.Name}</Name>

```

```

121     <Author>${newBook.Author}</Author>
122     <Price>${newBook.Price}</Price>
123   </newBook>
124 </AddNew>
125 </soap:Body>
126 </soap:Envelope>`;
127   axios.post(this.URI + '?op=AddNew', body, this.config).then((response) => {
128     var xmlData = response.data;
129     xml2js.parseString(xmlData, { explicitArray: false, ignoreAttrs: true }, (err, result) => {
130       var result = result['soap:Envelope']['soap:Body'].AddNewResponse.
131         AddNewResult;
132       if (result) {
133         alert('OK BABY!');
134         this.getAll();
135       } else {
136         alert('SORRY BABY!');
137       }
138     });
139   });
140   async update(newBook) {
141     var body = `<?xml version="1.0" encoding="utf-8"?>
142 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
143   http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap
144   /envelope/">
145 <soap:Body>
146   <Update xmlns="http://sonkk.org/">
147     <newBook>
148       <Code>${newBook.Code}</Code>
149       <Name>${newBook.Name}</Name>
150       <Author>${newBook.Author}</Author>
151       <Price>${newBook.Price}</Price>
152     </newBook>
153   </Update>
154 </soap:Body>
155 </soap:Envelope>`;
156   axios.post(this.URI + '?op=Update', body, this.config).then((response) => {
157     var xmlData = response.data;
158     xml2js.parseString(xmlData, { explicitArray: false, ignoreAttrs: true }, (err, result) => {
159       var result = result['soap:Envelope']['soap:Body'].UpdateResponse.
160         UpdateResult;
161       if (result) {
162         alert('OK BABY!');
163         this.getAll();
164       } else {
165         alert('SORRY BABY!');
166       }
167     });
168   });
169 }

```

```

166     }
167     async delete(code) {
168       var body = `<?xml version="1.0" encoding="utf-8"?>
169 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
170   <soap:Body>
171     <Delete xmlns="http://sonkk.org/">
172       <code>${code}</code>
173     </Delete>
174   </soap:Body>
175 </soap:Envelope>`;
176     axios.post(this.URI + '?op=Delete', body, this.config).then((response) => {
177       var xmlData = response.data;
178       xml2js.parseString(xmlData, { explicitArray: false, ignoreAttrs: true }, (err, result) => {
179         var result = result['soap:Envelope']['soap:Body'].DeleteResponse.DeleteResult;
180         if (result) {
181           alert('OK BABY!');
182           this.getAll();
183         } else {
184           alert('SORRY BABY!');
185         }
186       });
187     });
188   }
189 }
190 export default Book;
191
192 const styles = StyleSheet.create({
193   loading: { flex: 1, justifyContent: 'center' },
194   container: { flex: 1, marginTop: 40, marginLeft: 10, marginRight: 10,
195     marginBottom: 20, borderWidth: 1 },
196   line: { marginTop: 10, marginBottom: 10, height: 1, backgroundColor: 'black' },
197   bookList: { marginTop: 10, alignItems: 'center', height: 180 },
198   flatList: { justifyContent: 'center' },
199   rowItem: { marginTop: 10 },
200   bookDetails: { alignItems: 'center' },
201   input: { marginTop: 10, padding: 5, width: 250, height: 40, borderWidth: 1 },
202   buttons: { flexDirection: 'row', marginTop: 10 }
203 });

```

### 2.3.3 RESTful Service

- Create 4 new projects:
  - Service:
    - Project type: ASP.NET Web Application (.NET Framework)

- Project name: BookSite
- Solution name: RestfulService
- Empty project » Check on Web API
- Winform Client:
  - Project type: Windows Forms App (.NET Framework)
  - Project name: BookClient
  - Solution name: RestfulService
- HTML Client:
  - Project type: HTML + Javascript
  - Project name: HtmlClient
  - Solution name: RestfulService
- Mobile Client:
  - Project type: React Native
  - Project name: MobileClient
  - Solution name: RestfulService

## BookSite project

- Create BooksController.cs (Web API controller)

```

1 ...
2 public class BooksController : ApiController {
3     [HttpGet]
4     [Route("api/books")]
5     public List<Book> GetAll() {
6         List<Book> books = new BookDAO().SelectAll();
7         return books;
8     }
9     [HttpGet]
10    [Route("api/books/{code}")]
11    public Book GetDetails(int code) {
12        Book book = new BookDAO().SelectByCode(code);
13        return book;
14    }
15    [HttpGet]
16    [Route("api/books/search/{keyword}")]
17    public List<Book> Search(String keyword) {
18        List<Book> books = new BookDAO().SelectByKeyword(keyword);
19        return books;
20    }
21    [HttpPost]
22    [Route("api/books")]
23    public bool AddNew(Book newBook) {
24        bool result = new BookDAO().Insert(newBook);
25        return result;

```

```

26 }
27 [HttpPost]
28 [Route("api/books/{code}")]
29 public bool Update(int code, Book newBook) {
30     if (code != newBook.Code) return false;
31     bool result = new BookDAO().Update(newBook);
32     return result;
33 }
34 [HttpDelete]
35 [Route("api/books/{code}")]
36 public bool Delete(int code) {
37     bool result = new BookDAO().Delete(code);
38     return result;
39 }

```

- Run on localhost: Ctrl+F5 » https://localhost:port/api/books
- Deploy on GearHost: Right-click Project » Publish... » http://sonkk.gear.host/api/books

- Postman:

- GetAll
  - (GET) http://sonkk.gear.host/api/books
- GetDetails
  - (GET) http://sonkk.gear.host/api/books/1
- Search
  - (GET) http://sonkk.gear.host/api/books/search/a
- AddNew
  - (POST) http://sonkk.gear.host/api/books
  - Body (raw+JSON): { "Code": 4, "Name": "MAD", "Author": "HSU4", "Price": 444 }
- Update
  - (PUT) http://sonkk.gear.host/api/books/4
  - Body (raw+JSON): { "Code": 4, "Name": "ADM", "Author": "4HSU", "Price": 123 }
- Delete
  - (DELETE) http://sonkk.gear.host/api/books/4

## BookClient project

- Install Newtonsoft.Json (James Newton-King) library:

- Right-click project » Manage NuGet Packages...
- Browse tab » Search "Newtonsoft.Json" » Install

- Create Book.cs (DTO class)

```

1 ...
2 class Book {
3     public int Code { get; set; }
4     public String Name { get; set; }
5     public String Author { get; set; }
6     public int Price { get; set; }
7 }
```

- Create BookForm.cs (presentation class)

```

1 ...
2 public partial class BookForm : Form {
3     private void BookForm_Load(object sender, EventArgs e) {
4         List<Book> books = new BookBUS().GetAll();
5         ...
6     }
7     private void gridBook_SelectionChanged(object sender, EventArgs e) {
8         ...
9         Book book = new BookBUS().GetDetails(code);
10        ...
11    }
12
13    private void btnSearch_Click(object sender, EventArgs e) {
14        ...
15        List<Book> books = new List<Book>();
16        if (keyword.Length > 0) {
17            books = new BookBUS().Search(keyword);
18        } else {
19            books = new BookBUS().GetAll();
20        }
21        ...
22    }
23
24    private void btnAdd_Click(object sender, EventArgs e) {
25        ...
26        bool result = new BookBUS().AddNew(newBook);
27        if (result) {
28            List<Book> books = new BookBUS().GetAll();
29            ...
30        }
31        ...
32    }
33
34    private void btnUpdate_Click(object sender, EventArgs e) {
35        ...
36        bool result = new BookBUS().Update(newBook);
37        if (result) {
38            List<Book> books = new BookBUS().GetAll();
39            ...
40        }
41    }
42 }
```

```

40     }
41     ...
42 }
43
44 private void btnDelete_Click(object sender, EventArgs e) {
45     ...
46     bool result = new BookBUS().Delete(code);
47     if (result) {
48         List<Book> books = new BookBUS().GetAll();
49         ...
50     }
51     ...
52 }
53 }
```

- Create BookBUS.cs (business-logic class)

```

1 ...
2 using System.Net;
3 using Newtonsoft.Json;
4
5 class BookBUS {
6     String URI = "http://sonkk.gear.host/api/books";
7
8     public List<Book> GetAll() {
9         WebClient client = new WebClient();
10        String response = client.DownloadString(URI);
11        return JsonConvert.DeserializeObject<List<Book>>(response);
12    }
13    public Book GetDetails(int code) {
14        WebClient client = new WebClient();
15        String response = client.DownloadString(URI + "/" + code);
16        return JsonConvert.DeserializeObject<Book>(response);
17    }
18    public List<Book> Search(String keyword) {
19        WebClient client = new WebClient();
20        String response = client.DownloadString(URI + "/search/" + keyword);
21        return JsonConvert.DeserializeObject<List<Book>>(response);
22    }
23    public bool AddNew(Book newBook) {
24        String data = JsonConvert.SerializeObject(newBook);
25        WebClient client = new WebClient();
26        client.Headers[HttpRequestHeader.ContentType] = "application/json";
27        String response = client.UploadString(URI, "POST", data);
28        return bool.Parse(response);
29    }
30    public bool Update(Book newBook) {
31        String data = JsonConvert.SerializeObject(newBook);
32        WebClient client = new WebClient();
33        client.Headers[HttpRequestHeader.ContentType] = "application/json";
34        String response = client.UploadString(URI + "/" + newBook.Code, "PUT", data);
```

```

35     return bool.Parse(response);
36 }
37 public bool Delete(int code) {
38     WebClient client = new WebClient();
39     String response = client.UploadString(URI + "/" + code, "DELETE", "");
40     return bool.Parse(response);
41 }
42 }
```

## HtmlClient project

- Create books.html (presentation page)

```

1 <html>
2 <head>
3   <script src="https://unpkg.com/axios/dist/axios.min.js"></script>
4   <script src="books.js"></script>
5 </head>
6 <body onload="page_Load()">
7 ...
8 </body>
9 </html>
```

- Create books.js (event-handlers)

```

1 const CORS = "https://cors-anywhere.herokuapp.com/";
2 const URI = "http://sonkk.gear.host/api/books";
3
4 /* event-handler methods */
5 ...
6 /* fetch API methods */
7 function getAll() {
8     axios.get(CORS + URI).then((response) => {
9         var books = response.data;
10        renderBookList(books);
11    });
12 }
13 function getDetails(code) {
14     axios.get(CORS + URI + "/" + code).then((response) => {
15         var book = response.data;
16         renderBookDetails(book);
17    });
18 }
19 function search(keyword) {
20     axios.get(CORS + URI + "/search/" + keyword).then((response) => {
21         var books = response.data;
22         renderBookList(books);
23    });
24 }
25 function addNew(newBook) {
26     axios.post(CORS + URI, newBook).then((response) => {
```

```

27     var result = response.data;
28     if (result) {
29         getAll();
30         clearTextboxes();
31     } else {
32         alert('SORRY BABY!');
33     }
34 });
35 }
36 function update(newBook) {
37     axios.put(CORS + URI + "/" + newBook.Code, newBook).then((response) => {
38         var result = response.data;
39         if (result) {
40             getAll();
41             clearTextboxes();
42         } else {
43             alert('SORRY BABY!');
44         }
45     });
46 }
47 function delete(code) {
48     axios.delete(CORS + URI + "/" + code).then((response) => {
49         var result = response.data;
50         if (result) {
51             getAll();
52             clearTextboxes();
53         } else {
54             alert('SORRY BABY!');
55         }
56     });
57 }
58 /* helper methods */
59 ...

```

## MobileClient project

- Create React Native project by Expo-cli

```

1 expo init MobileClient
2 cd MobileClient
3 expo start -c

```

- Update App.js (entry-point)

```

1 import React, { Component } from 'react';
2 import Book from './components/BookComponent';
3
4 class App extends Component {
5     render() {
6         return (<Book />);
7     }

```

```

8 }
9 export default App;

```

- Create components/BookComponent.js

```

1 import React, { Component } from 'react';
2 import { StyleSheet, View, ActivityIndicator, FlatList, TouchableOpacity, Text,
3   TextInput, Button, Alert } from 'react-native';
4
5 class Book extends Component {
6   constructor(props) {
7     super(props);
8     this.URI = 'http://sonkk.gear.host/api/books';
9     this.state = { isLoading: true, dataSource: [], txtCode: '', txtName: '',
10       txtAuthor: '', txtPrice: '' };
11   }
12   render() {
13     ...
14   }
15   componentDidMount() {
16     ...
17   }
18   /* event-handler methods */
19   pressOnRow(item) {
20     ...
21   }
22   btnAdd = async () => {
23     ...
24   }
25   btnUpdate = () => {
26     ...
27   }
28   btnDelete = () => {
29     ...
30   }
31   /* fetch API methods */
32   async getAll() {
33     var response = await fetch(this.URI);
34     var books = await response.json();
35     this.setState({ isLoading: false, dataSource: books });
36   }
37   async getDetails(code) {
38     var response = await fetch(this.URI + '/' + code);
39     var book = await response.json();
40     this.setState({ txtCode: book.Code.toString(), txtName: book.Name, txtAuthor:
41       book.Author, txtPrice: book.Price.toString() });
42   }
43   async addNew(newBook) {
44     var response = await fetch(this.URI, { method: 'POST', headers: { 'Content-
45       Type': 'application/json' }, body: JSON.stringify(newBook) });
46     var result = await response.json();

```

```

43     if (result) {
44         alert('OK BABY!');
45         this.getAll();
46     } else {
47         alert('SORRY BABY!');
48     }
49 }
50 async update(newBook) {
51     var response = await fetch(this.URI + '/' + newBook.Code, { method: 'PUT',
52         headers: { 'Content-Type': 'application/json' }, body: JSON.stringify(
53             newBook) });
54     var result = await response.json();
55     if (result) {
56         alert('OK BABY!');
57         this.getAll();
58     } else {
59         alert('SORRY BABY!');
60     }
61 }
62 async delete(code) {
63     var response = await fetch(this.URI + '/' + code, { method: 'DELETE' });
64     var result = await response.json();
65     if (result) {
66         alert('OK BABY!');
67         this.getAll();
68     } else {
69         alert('SORRY BABY!');
70     }
71 }
72 export default Book;
73
74 const styles = StyleSheet.create({
75     ...
76 });

```

#### 2.3.4 Backend as a Service

- Create 3 new projects:

- Winform Client:
  - Project type: Windows Forms App (.NET Framework)
  - Project name: BookClient
  - Solution name: BaaS
- HTML Client:
  - Project type: HTML + Javascript
  - Project name: HtmlClient
  - Solution name: BaaS

- Mobile Client:

- Project type: React Native
- Project name: MobileClient
- Solution name: BaaS

### BookClient project

- Install FireSharp (ziyasal) library:

- Right-click project » Manage NuGet Packages...
- Browse tab » Search "FireSharp" » Install

- Create Book.cs (DTO class)

```

1 ...
2 class Book {
3     public int Code { get; set; }
4     public String Name { get; set; }
5     public String Author { get; set; }
6     public int Price { get; set; }
7 }
```

- Create BookForm.cs (presentation class)

```

1 ...
2 public partial class BookForm : Form {
3     private void BookForm_Load(object sender, EventArgs e) {
4         new BookBUS().ListenFirebase(gridBook);
5     }
6     private void gridBook_SelectionChanged(object sender, EventArgs e) {
7         ...
8         Book book = new BookBUS().GetDetails(code);
9         ...
10    }
11    private void btnSearch_Click(object sender, EventArgs e) {
12        ...
13        List<Book> books = new BookBUS().Search(keyword);
14        gridBook.BeginInvoke(new MethodInvoker(delegate { gridBook.DataSource =
15            books; })); // set asynchronous datasource
16    }
17    private void btnAdd_Click(object sender, EventArgs e) {
18        ...
19        bool result = new BookBUS().AddNew(newBook);
20        if (!result) MessageBox.Show("SORRY BABY !!!");
21    }
22    private void btnUpdate_Click(object sender, EventArgs e) {
23        ...
24        bool result = new BookBUS().Update(newBook);
25        if (!result) MessageBox.Show("SORRY BABY !!!");
```

```

25    }
26    private void btnDelete_Click(object sender, EventArgs e) {
27      ...
28      bool result = new BookBUS().Delete(code);
29      if (!result) MessageBox.Show("SORRY BABY !!!");
30    }
31 }

```

- Create BookBUS.cs (business-logic class)

```

1 ...
2 using FireSharp;
3 using FireSharp.Config;
4 using FireSharp.Interfaces;
5 using FireSharp.Response;
6
7 class BookBUS {
8   static IFirebaseConfig config = new FirebaseConfig { BasePath = "https://sonkk
9     .firebaseio.com" };
10  static FirebaseClient client = new FirebaseClient(config);
11
12  public async void ListenFirebase(DataGridView gridBook) {
13    EventStreamResponse response = await client.OnAsync("books",
14      added: (sender, args, context) => { UpdateDataGridView(gridBook); },
15      changed: (sender, args, context) => { UpdateDataGridView(gridBook); },
16      removed: (sender, args, context) => { UpdateDataGridView(gridBook); }
17    );
18  }
19  private void UpdateDataGridView(DataGridView gridBook) {
20    List<Book> books = GetAll();
21    gridBook.BeginInvoke(new MethodInvoker(delegate { gridBook.DataSource =
22      books; })); // set asynchronous datasource
23  }
24  private List<Book> GetAll() {
25    FirebaseResponse response = client.Get("books");
26    Dictionary<String, Book> dictBooks = response.ResultAs<Dictionary<String,
27      Book>>();
28    return dictBooks.Values.ToList();
29  }
30  public Book GetDetails(int code) {
31    FirebaseResponse response = client.Get("books/B" + code);
32    Book book = response.ResultAs<Book>();
33    return book;
34  }
35  private String GetKeyByCode(int code) {
36    FirebaseResponse response = client.Get("books");
37    Dictionary<String, Book> dictBooks = response.ResultAs<Dictionary<String,
38      Book>>();
39    String key = dictBooks.FirstOrDefault(x => x.Value.Code == code).Key;
40    return key;
41  }

```

```

38     public List<Book> Search(String keyword) {
39         List<Book> books = new List<Book>();
40         foreach (var item in GetAll()) {
41             if (item.Name.ToLower().Contains(keyword.ToLower())) {
42                 books.Add(item);
43             }
44         }
45         return books;
46     }
47     public bool AddNew(Book newBook) {
48         try {
49             //client.Push("books", newBook); // auto-generated key
50             client.Set("books/B" + newBook.Code, newBook); // custom key
51             return true;
52         } catch { return false; }
53     }
54     public bool Update(Book newBook) {
55         try {
56             String key = GetKeyByCode(newBook.Code);
57             if (String.IsNullOrEmpty(key)) return false;
58             client.Set("books/" + key, newBook);
59             return true;
60         } catch { return false; }
61     }
62     public bool Delete(int code) {
63         try {
64             String key = GetKeyByCode(code);
65             if (String.IsNullOrEmpty(key)) return false;
66             client.Delete("books/" + key);
67             return true;
68         } catch { return false; }
69     }
70 }
```

## HtmlClient project

- Create books.html (presentation page)

```

1 <html>
2 <head>
3     <script src="https://www.gstatic.com/firebasejs/8.3.1.firebaseio.js"></script>
4     <script src="books.js"></script>
5 </head>
6 <body onload="page_Load()">
7     ...
8 </body>
9 </html>
```

- Create books.js (event-handlers)

```
1 const config = { databaseURL: "https://sonkk.firebaseio.com" };
```

```

2  firebase.initializeApp(config);
3  const dbRef = firebase.database().ref();
4
5  /* event-handler methods */
6  function btnAdd_Click() {
7    var newBook = {
8      Code: document.getElementById("txtCode").value,
9      Name: document.getElementById("txtName").value,
10     Author: document.getElementById("txtAuthor").value,
11     Price: document.getElementById("txtPrice").value
12   };
13   addNew(newBook);
14 }
15 ...
16 /* firebase methods */
17 function getAll() {
18   dbRef.child("books").on("value", (snapshot) => {
19     var books = [];
20     snapshot.forEach((child) => {
21       //alert(child.key);
22       var book = child.val();
23       books.push(book);
24     });
25     renderBookList(books);
26   });
27 }
28 function getDetails(code) {
29   dbRef.child("books").once("value", (snapshot) => {
30     snapshot.forEach((child) => {
31       var book = child.val();
32       if (book.Code == code) {
33         renderBookDetails(book);
34       }
35     });
36   });
37 }
38 function search(keyword) {
39   dbRef.child("books").once("value", (snapshot) => {
40     var books = [];
41     snapshot.forEach((child) => {
42       var book = child.val();
43       if (book.Name.toLowerCase().includes(keyword.toLowerCase())) {
44         books.push(book);
45       }
46     });
47     renderBookList(books);
48   });
49 }
50 function addNew(newBook) {
51   //dbRef.child("books").push(newBook); // auto-generated key
52   dbRef.child("books/B" + newBook.Code).set(newBook); // custom key

```

```

53 }
54 function update(newBook) {
55   dbRef.child("books").once("value", (snapshot) => {
56     snapshot.forEach((child) => {
57       var book = child.val();
58       if (book.Code == newBook.Code) {
59         var key = child.key;
60         dbRef.child("books").child(key).set(newBook);
61       }
62     });
63   });
64 }
65 function delete(code) {
66   dbRef.child("books").once("value", (snapshot) => {
67     snapshot.forEach((child) => {
68       var book = child.val();
69       if (book.Code == code) {
70         var key = child.key;
71         dbRef.child("books").child(key).remove();
72       }
73     });
74   });
75 }
76 /* helper methods */
77 ...

```

## MobileClient project

- Create React Native project by Expo-cli

```

1 expo init MobileClient
2 cd MobileClient
3 npm install firebase --save
4 expo start -c

```

- Update App.js (entry-point)

```

1 import React, { Component } from 'react';
2 import Book from './components/BookComponent';
3
4 class App extends Component {
5   render() {
6     return (<Book />);
7   }
8 }
9 export default App;

```

- Create components/BookComponent.js

```

1 import React, { Component } from 'react';

```

```

2 import { StyleSheet, View, ActivityIndicator, FlatList, TouchableOpacity, Text,
3         TextInput, Button, Alert } from 'react-native';
4 import firebase from 'firebase';
5
6 class Book extends Component {
7   constructor(props) {
8     super(props);
9     this.state = { isLoading: true, dataSource: [], txtCode: '', txtName: '',
10                  txtAuthor: '', txtPrice: '' };
11   }
12   render() {
13     ...
14     <Text>BOOK DETAILS</Text>
15     <TextInput style={styles.input} underlineColorAndroid='transparent'
16               placeholder='Code' autoCapitalize='none' value={this.state.txtCode}
17               onChangeText={(txtCode) => this.setState({ txtCode })} />
18     ...
19   }
20   /* event-handler methods */
21   pressOnRow(item) {
22     ...
23   }
24   btnAdd = async () => {
25     var newBook = { Code: this.state.txtCode, Name: this.state.txtName, Author:
26                   this.state.txtAuthor, Price: this.state.txtPrice };
27     this.addNew(newBook);
28   }
29   btnUpdate = () => {
30     ...
31   }
32   btnDelete = () => {
33     ...
34   }
35   /* firebase methods */
36   initFirebase() {
37     const config = { databaseURL: 'https://sonkk.firebaseio.com' };
38     firebase.initializeApp(config);
39     this.dbRef = firebase.database().ref();
40   }
41   getAll() {
42     this.dbRef.child("books").on("value", (snapshot) => {
43       var books = [];
44       snapshot.forEach((child) => {
45         var book = child.val();
46         books.push(book);
47       });
48       this.setState({ isLoading: false, dataSource: books });
49     });
50   }
51 }
52 
```

```

48     });
49 }
50 getDetails(code) {
51   this.dbRef.child("books").once("value", (snapshot) => {
52     snapshot.forEach((child) => {
53       var book = child.val();
54       if (book.Code == code) {
55         this.setState({ txtCode: book.Code.toString(), txtName: book.Name,
56                       txtAuthor: book.Author, txtPrice: book.Price.toString() });
57       }
58     });
59   });
60 addNew(newBook) {
61   //this.dbRef.child("books").push(newBook); // auto-generated key
62   this.dbRef.child("books/B" + newBook.Code).set(newBook); // custom key
63 }
64 update(newBook) {
65   this.dbRef.child("books").once("value", (snapshot) => {
66     snapshot.forEach((child) => {
67       var book = child.val();
68       if (book.Code == newBook.Code) {
69         var key = child.key;
70         this.dbRef.child("books").child(key).set(newBook);
71       }
72     });
73   });
74 }
75 delete(code) {
76   this.dbRef.child("books").once("value", (snapshot) => {
77     snapshot.forEach((child) => {
78       var book = child.val();
79       if (book.Code == code) {
80         var key = child.key;
81         this.dbRef.child("books").child(key).remove();
82       }
83     });
84   });
85 }
86 }
87 export default Book;
88
89 const styles = StyleSheet.create({
90   ...
91 });

```

### 2.3.5 Microservice

- Create 2 new projects:

- Service:

- Project type: ASP.NET Web Application (.NET Framework)
- Project name: BookSite
- Solution name: Microservice
- Empty project » Check on Web API
- Winform Client:
  - Project type: Windows Forms App (.NET Framework)
  - Project name: BookClient
  - Solution name: Microservice

### **BookClient project**

- Install Newtonsoft.Json (James Newton-King) & FireSharp (ziyasal) libraries
- Update Program.cs (entry-point)

```

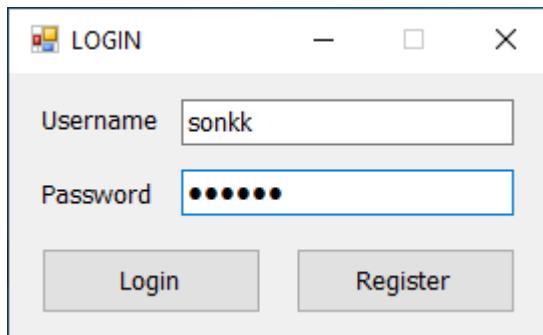
1 ...
2 static class Program {
3     [STAThread]
4     static void Main() {
5         ...
6         Application.Run(new LoginForm());
7     }
8 }
```

- Create Account.cs (DTO class)

```

1 ...
2 class Account {
3     public String Username { get; set; }
4     public String Password { get; set; }
5 }
```

- Create LoginForm.cs (presentation class)



```

1 ...
2 public partial class LoginForm : Form {
3     ...
4     private void btnRegister_Click(object sender, EventArgs e) {
```

```

5     Account newAccount = new Account() {
6         Username = txtUsername.Text.Trim(),
7         Password = txtPassword.Text.Trim()
8     };
9     bool result = new AccountBUS().AddNew(newAccount);
10    if (result) {
11        MessageBox.Show("OK BABY!");
12    } else {
13        MessageBox.Show("SORRY BABY!");
14    }
15}
16 private void btnLogin_Click(object sender, EventArgs e) {
17     Account account = new Account() {
18         Username = txtUsername.Text.Trim(),
19         Password = txtPassword.Text.Trim()
20     };
21     bool result = new AccountBUS().CheckAccount(account);
22     if (result) {
23         new BookForm().Show();
24         this.Hide();
25     } else {
26         MessageBox.Show("SORRY BABY!");
27     }
28 }
29}

```

- Create AccountBUS.cs (business-logic class)

```

1 ...
2 using FireSharp;
3 using FireSharp.Config;
4 using FireSharp.Interfaces;
5 using FireSharp.Response;
6
7 class AccountBUS {
8     static IFirebaseConfig config = new FirebaseConfig { BasePath = "https://sonkk
9         .firebaseio.com" };
10    static FirebaseClient client = new FirebaseClient(config);
11
12    public bool AddNew(Account newAccount) {
13        try {
14            client.Push("accounts", newAccount); // auto-generated key
15            return true;
16        }
17        catch { return false; }
18    }
19    public bool CheckAccount(Account account) {
20        FirebaseResponse response = client.Get("accounts");
21        Dictionary<String, Account> dicAccounts = response.ResultAs<Dictionary<
22            String, Account>>();
23        String key = dicAccounts.FirstOrDefault(x => x.Value.Username == account.

```

```
        Username && x.Value.Password == account.Password).Key;
22    if (String.IsNullOrEmpty(key))
23        return false;
24    return true;
25 }
26 }
```

## 2.4 Others architecture

### 2.4.1 MVC

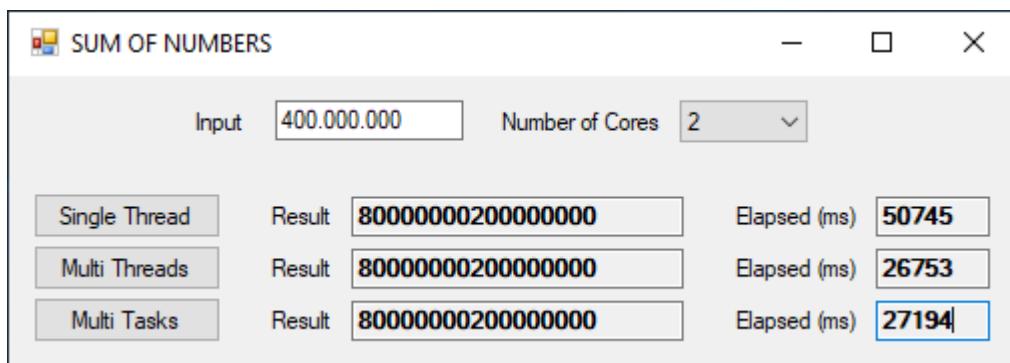
- ASP.NET MVC 5: <http://www.asp.net/mvc/tutorials/mvc-5/introduction/getting-started>
  - ASP.NET Core MVC: <http://docs.asp.net/en/latest/tutorials/first-mvc-app>

### 2.4.2 Parallel

- Create a new project:
    - Project type: Windows Forms App (.NET Framework)
    - Project name: SumOfNumbers
    - Solution name: Parallel

## SumOfNumbers project

- Create SumForm.cs (presentation class)



```
1 ...
2 using System.Numerics; // add reference System.Numerics.dll
3 using System.Diagnostics;
4 using System.Threading;
5
6 public partial class SumForm : Form {
7     public SumForm() {
8         InitializeComponent();
9         cmbCores.SelectedIndex = 0;
10    }
11    private void btnSingleThread_Click(object sender, EventArgs e) {
12        btnSingleThread.Enabled = false;
13        txtResult1.Text = "";
```

```

14     txtElapsed1.Text = "";
15     // get input
16     int input = int.Parse(txtInput.Text.Replace(".", "").Trim());
17     // start calculate
18     Stopwatch watch = Stopwatch.StartNew();
19     SumNumbers worker = new SumNumbers(1, input);
20     worker.Calculate();
21     BigInteger result = worker.Result;
22     watch.Stop();
23     long elapsed = watch.ElapsedMilliseconds;
24     // display result
25     txtResult1.Text = result.ToString();
26     txtElapsed1.Text = elapsed.ToString();
27     btnSingleThread.Enabled = true;
28 }
29 private void btnMultiThreads_Click(object sender, EventArgs e) {
30     btnMultiThreads.Enabled = false;
31     txtResult2.Text = "";
32     txtElapsed2.Text = "";
33     // get input
34     int input = int.Parse(txtInput.Text.Replace(".", "").Trim());
35     int numCores = int.Parse(cmbCores.SelectedItem.ToString().Trim());
36     // assign tasks
37     Thread[] threads = new Thread[numCores];
38     SumNumbers[] workers = new SumNumbers[numCores];
39     for (int i = 1; i <= numCores; i++) {
40         int from = (i == 1) ? 1 : (input / numCores * (i - 1)) + 1;
41         int to = (i == numCores) ? input : input / numCores * i;
42         workers[i - 1] = new SumNumbers(from, to);
43         threads[i - 1] = new Thread(new ThreadStart(workers[i - 1].Calculate));
44     }
45     // start calculate
46     Stopwatch watch = Stopwatch.StartNew();
47     foreach(Thread thread in threads) {
48         thread.Start();
49     }
50     foreach(Thread thread in threads) {
51         thread.Join();
52     }
53     // collect result
54     BigInteger result = new BigInteger(0);
55     foreach(SumNumbers worker in workers) {
56         result += worker.Result;
57     }
58     watch.Stop();
59     long elapsed = watch.ElapsedMilliseconds;
60     // display result
61     txtResult2.Text = result.ToString();
62     txtElapsed2.Text = elapsed.ToString();
63     btnMultiThreads.Enabled = true;
64 }
```

```

65  private void btnMultiTasks_Click(object sender, EventArgs e) {
66      btnMultiTasks.Enabled = false;
67      txtResult3.Text = "";
68      txtElapsed3.Text = "";
69      // get input
70      int input = int.Parse(txtInput.Text.Replace(".", "").Trim());
71      int numCores = int.Parse(cmbCores.SelectedItem.ToString().Trim());
72      // assign tasks
73      Task[] tasks = new Task[numCores];
74      SumNumbers[] workers = new SumNumbers[numCores];
75      for (int i = 1; i <= numCores; i++) {
76          int from = (i == 1) ? 1 : (input / numCores * (i - 1)) + 1;
77          int to = (i == numCores) ? input : input / numCores * i;
78          workers[i - 1] = new SumNumbers(from, to);
79          tasks[i - 1] = new Task(new Action(workers[i - 1].Calculate));
80      }
81      // start calculate
82      Stopwatch watch = Stopwatch.StartNew();
83      foreach (Task task in tasks) {
84          task.Start();
85      }
86      Task.WaitAll(tasks);
87      // collect result
88      BigInteger result = new BigInteger(0);
89      foreach (SumNumbers worker in workers) {
90          result += worker.Result;
91      }
92      watch.Stop();
93      long elapsed = watch.ElapsedMilliseconds;
94      // display result
95      txtResult3.Text = result.ToString();
96      txtElapsed3.Text = elapsed.ToString();
97      btnMultiTasks.Enabled = true;
98  }
99 }
```

- Create SumNumbers.cs (business-logic class)

```

1 ...
2 using System.Numerics; // add reference System.Numerics.dll
3
4 class SumNumbers {
5     public int From { get; set; }
6     public int To { get; set; }
7     public BigInteger Result { get; set; }
8
9     public SumNumbers(int from, int to) {
10        this.From = from;
11        this.To = to;
12        this.Result = new BigInteger(0);
13    }
14 }
```

```
14 public void Calculate() {  
15     for (int i = this.From; i <= this.To; i++) {  
16         this.Result += new BigInteger(i);  
17     }  
18 }  
19 }
```