Name :- Nisarg .K. Amlani
Roll :- CE001
ID :- 22ceueg082

Lab 1

Q1) What is monolithic architecture? Mention the advantages and disadvantages.

Ans) Monolithic Architecture is a design pattern that is combined of an application's component into a single inseparable unit .

Advantages

- **Simplicity** : Monolithic architecture is easy to develop, test, and deploy because it's built as a single unit with a single codebase.
- **Cost-Effective :** Monolithic architecture can be cost-effective to start with, requiring less investment in infrastructure and human resources.
- **Specialist knowledge**: As an application grows, development teams can specialize in one or two parts, allowing technology specialists to apply their knowledge.

Disadvantages

- **Scalability :** It's difficult to scale individual parts of a monolithic application because all components are bundled together.
- **Updates and maintenance :** Monolithic applications are more complex to update and maintain than microservices-based applications.
- **Single point of failure**: Monolithic architecture has a single point of failure, which can increase the risk of system-wide outages.

- **Continuous deployment**: Monolithic architecture can be challenging for continuous deployment because the entire application must be redeployed for any change.

Q2) What is client - server architecture ? mention advantages and disadvantages ?

Ans) The Client-server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters called clients

Advantages

- Centralized system with all data in a single place.
- Cost efficient requires less maintenance cost and Data recovery is possible.
- The capacity of the Client and Servers can be changed separately.

Disadvantages

- Clients are prone to viruses, Trojans, and worms if present in the Server or uploaded into the Server.
- Servers are prone to Denial of Service (DOS) attacks.
- Data packets may be spoofed or modified during transmission.

Q3) What is the difference between single-tier and two-tier client server architectures? Mention the applications of both.

Ans) **Single-tier client-server architecture,** also known as one-tier architecture, is a software architecture where all the components of an application are stored on a single device or shared storage device.

Typically used for small , simple application a prototype

A **two-tier client-server architecture** is an IT infrastructure model that separates application components into two layers: the client or front-end, and the server or back-end. The client interacts directly with the server through a protocol.

E.g Used for non-complex , non-time critical information system

Q4) What is multi-tier client server architecture? Mention the advantages and applications.

Ans) A software architecture that organizes application components into tiers, or layers, to provide specific functionality.

Advantages

- **Improved scalability :** Each tier can be scaled independently of the others, allowing you to meet changing performance demands.
- **Easy Maintenance :** Modifications to one tier often have minimal or no effect on other tiers.
- **Easier Operations :** In a large system, only a few people are expert at each tier's applications .

Application : provides a general framework to ensure decoupled and independently scalable application components can be separately developed, managed, and maintained

Q5) What is distributed internet architecture? Compare and contrast with other architecture and identify the applications.

Ans) A distributed internet architecture is a system where components are spread across multiple interconnected computers or servers.

Comparison with other Architectures

Centralized Architecture:

- **Single point of failure:** A single server handles all requests, making it vulnerable to failures.
- **Scalability limitations:** Difficulty in scaling to handle increased load.
- **Lower fault tolerance:** A failure in the central server can disrupt the entire system.

Client-Server Architecture:

- **Improved scalability:** Multiple servers can handle requests, increasing capacity.
- **Better fault tolerance:** Failure of one server can be mitigated by others.
- **Still relies on a central server:** A single point of failure remains, albeit less critical.

Distributed Architecture:

- **High scalability:** Effortlessly scales by adding more nodes.
- **Enhanced fault tolerance:** Failure of one node does not affect the overall system.
- **Complex management:** Requires sophisticated management and coordination.

Q6) What is hybrid web-services architecture? Identify the difference with other architectures and mention the applications.

Ans) Hybrid web-service architecture combines multiple architecture style in one. Often combining RESTful APIs , microservices and other Arch.

Difference with other architectures

- **Traditional Web Services (SOAP):** More complex and verbose, often requiring extensive configuration.

- **RESTful APIs:** Simpler and more lightweight, but may lack the rich feature set of SOAP.
- **Microservices:** Highly modular, but can be complex to manage and deploy.
- **Applications**
- **Enterprise Integration:** Integrating diverse systems and applications within an organization.
- **API-Driven Development:** Creating APIs for internal and external use

Q7) What is microservice architecture? Identify the need and applications.

Ans) A microservice architecture is a design pattern where a single application is broken down into smaller, independent services. Each service focuses on a specific business capability and communicates with others through well-defined APIs.

Need for Microservice Architecture:

- **Scalability:** Independent scaling of services based on their specific needs.
- **Flexibility**: Easier to adopt new technologies and frameworks for different services.
- **Resilience**: Isolated failures, minimizing impact on the overall system.

Applications of Microservice Architecture:

- **E-commerce:** Product catalog, order processing, payment processing, and shipping.
- **Financial Services:** Account management, trading, risk assessment, and fraud detection.
- **Social Media:** User profiles, news feeds, messaging, and notifications.

Q8) Identify and list the differences between Client-Server architecture and Service Oriented Architecture.

Ans)

Client-Server Architecture

- **Centralized Server:** A single server handles requests from multiple clients.
- **Direct Communication:** Clients communicate directly with the server.
- **Simple Structure:** Relatively straightforward architecture.
- **Limited Scalability:** Can be scaled by adding more servers, but it can be complex.
- **Tight Coupling:** Clients and server are tightly coupled, making changes to one often requiring changes to the other.

Service-Oriented Architecture (SOA)

- **Decentralized Services:** Multiple services, each with a specific function.
- **Loose Coupling:** Services communicate through well-defined interfaces, reducing dependencies.
- **Reusability:** Services can be reused across multiple applications.
- **Flexibility:** Easier to add, remove, or modify services without affecting the entire system.
- **Scalability:** Can be scaled by adding more instances of services or by using load balancing.
- **Complex Implementation:** Requires careful design and management.

Q9) Identify and list the differences between Distributed Internet architecture and Service Oriented Architecture.

Ans) Distributed Internet Architecture

- **Focus**: Network infrastructure and data distribution.
- **Granularity**: Components are typically smaller, focused on specific tasks.
- **Communication**: Relies on network protocols (HTTP, TCP/IP) for direct communication.
- **Coupling**: Components are loosely coupled, often using APIs or message queues.
- **Scalability**: Highly scalable, as components can be added or removed independently.
- **Fault Tolerance**: Built-in fault tolerance through redundancy, replication, and failover mechanisms.

Service-Oriented Architecture (SOA)

- **Focus**: Business processes and services.
- **Granularity**: Services are typically larger, encapsulating significant business capabilities.
- **Communication**: Often relies on a centralized Enterprise Service Bus (ESB) for message routing and transformation.
- **Coupling**: Services can be loosely coupled, but often rely on shared resources or infrastructure.
- **Scalability**: Can be scaled, but may have limitations.
- **Fault Tolerance**: Can be designed to be fault-tolerant, but often relies on the ESB for handling failures.

Q10) Identify and list the differences between Hybrid web-services architecture and Service Oriented Architecture.

Ans)

Hybrid Web Services Architecture

- **Combination of Styles:** Leverages a mix of architectural styles, such as SOAP, REST, and GraphQL.
- **Flexibility:** Can adapt to different use cases and evolving technologies.
- **Scalability:** Can scale both horizontally and vertically to accommodate varying workloads.
- **Performance:** Can optimize performance through caching, load balancing, and other techniques.
- **Security:** Implements robust security measures to protect sensitive data.

Service-Oriented Architecture (SOA)

- **Focus:** Business processes and services.
- **Granularity:** Services are typically larger, encapsulating significant business capabilities.
- **Communication:** Often relies on a centralized Enterprise Service Bus (ESB) for message routing and transformation.
- **Coupling:** Services can be loosely coupled, but often rely on shared resources or infrastructure.
- **Scalability:** Can be scaled, but may have limitations.
- **Fault Tolerance:** Can be designed to be fault-tolerant, but often relies on the ESB for handling failures.

Q11) Identify and list the primary differences between SOAP based and RESTful web services.

Ans)

SOAP (Simple Object Access Protocol)

- **XML-based:** Uses XML for both message format and protocol.
- **WSDL:** Requires a WSDL (Web Services Description Language) to define the service contract.
- **Stateful or Stateless:** Can be stateful or stateless.

- **Security:** Built-in security mechanisms like WS-Security.
- **Complexity:** More complex to implement and understand.
- **Performance:** Can be less performant due to XML parsing overhead.

REST (Representational State Transfer)

- **Resource-Based:** Treats data as resources identified by URIs.
- **Stateless:** Each request is independent, without server-side session state.
- **HTTP Methods:** Uses standard HTTP methods (GET, POST, PUT, DELETE) for operations.
- **JSON or XML:** Can use JSON or XML for data format.
- **Simplicity:** Simpler to implement and understand.
- **Performance:** Generally more performant due to lightweight protocols.

Name :- Nisarg Amlani
Roll :- CE001
ID :- 22ceueg082
Lab :- 2

=>Xml file content

```
<?xml version="1.0" encoding="utf-8" ?>
<?xml-stylesheet type="text/xsl" href="library_transform.xsl"?>
<library>
  <categories id="cat-1">
    <name>
      Fiction
    </name>
    <books>
      <book id="b-001">
        <title>1984</title>
        <bookAuthor id="A101">George Orwell</bookAuthor>
        <publication year="1949"/>
        <price currency="USD">9.99</price>
      </book>
      <book id="b-002">
        <title>To Kill a Mockingbird</title>
        <bookAuthor id="A102">Harper Lee</bookAuthor>
        <publication year="1960"/>
        <price currency="USD">7.99</price>
      </book>
    </books>
  </categories>
  <categories id="cat-2">
    <name>
      Science
    </name>
    <books>
      <book id="b-003">
        <title>A Brief History of Time</title>
```

```
            <bookAuthor id="A103">Stephen Hawking</bookAuthor>
            <publication year="1988"/>
            <price currency="USD">15.00</price>
        </book>
    </books>
  </categories>
  <authors>
    <author id="A101">
      <authorName>George Orwell</authorName>
      <nationality>British</nationality>
    </author>
    <author id="A102">
      <authorName>Harper Lee</authorName>
      <nationality>American</nationality>
    </author>
    <author id="A103">
      <authorName>Stephen Hawking</authorName>
      <nationality>British</nationality>
    </author>
  </authors>
</library>
```

=>Xsl file content

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <!-- Output settings -->
  <xsl:output method="html" indent="yes" />
  <!-- Root template -->

  <xsl:template match="/library">
    <html>
      <head>
        <title>Library Collection</title>
        <style>
          body { font-family: Arial, sans-serif; margin: 20px; }
          h1 { color: #2E8B57; }
          h2 { color: #4682B4; margin-top: 20px; }
          table { border-collapse: collapse; width: 100%; margin-top: 10px; }
          th, td { border: 1px solid #ddd; padding: 8px; text-align: left; }
```

```
            th { background-color: #f2f2f2; }
        </style>
    </head>
    <body>
        <h1>Library Collection</h1>
        <!-- Categories and Books -->
        <xsl:apply-templates select="categories" />
        <!-- Author Information -->
        <h2>Authors</h2>
        <table>
            <tr>
                <th>Author ID</th>
                <th>Name</th>
                <th>Nationality</th>
            </tr>
            <xsl:apply-templates select="authors/author" />
        </table>
    </body>
  </html>
</xsl:template>

<!-- Template for Categories -->
<xsl:template match="categories">
  <h2>Category: <xsl:value-of select="name" /></h2>
  <table>
    <tr>
      <th>Book ID</th>
      <th>Title</th>
      <th>Author</th>
      <th>Publication Year</th>
      <th>Price (Currency)</th>
    </tr>
    <xsl:apply-templates select="books/book" />
  </table>
</xsl:template>

<!-- Template for Books -->
<xsl:template match="book">
  <tr>
    <td><xsl:value-of select="@id" /></td>
```

```
        <td><xsl:value-of select="title" /></td>
        <!-- Fetch Author Name using IDREF -->
        <td>
            <xsl:value-of
select="/library/authors/author[@id=current()/bookAuthor/@id]/authorName" />
        </td>
        <td><xsl:value-of select="publication/@year" /></td>
        <td><xsl:value-of select="price" /> (<xsl:value-of select="price/@currency"
/>)</td>
      </tr>
   </xsl:template>


   <!-- Template for Authors -->
   <xsl:template match="author">
      <tr>
         <td><xsl:value-of select="@id" /></td>
         <td><xsl:value-of select="authorName" /></td>
         <td><xsl:value-of select="nationality" /></td>
      </tr>
   </xsl:template>
</xsl:stylesheet>
```

=>Output of xml on browser

Library Collection

Category: Fiction

Book ID	Title	Author	Publication Year	Price (Currency)
b-001	1984	George Orwell	1949	9.99 (USD)
b-002	To Kill a Mockingbird	Harper Lee	1960	7.99 (USD)

Category: Science

Book ID	Title	Author	Publication Year	Price (Currency)
b-003	A Brief History of Time	Stephen Hawking	1988	15.00 (USD)

Authors

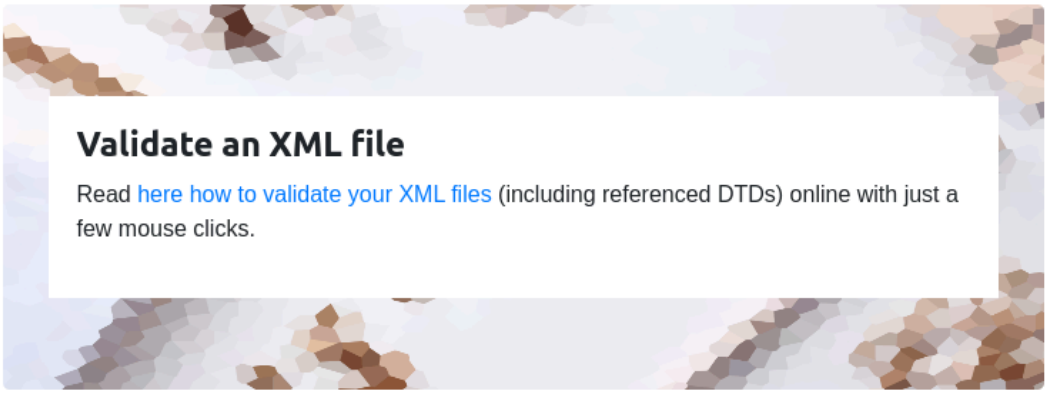
Author ID	Name	Nationality
A101	George Orwell	British
A102	Harper Lee	American
A103	Stephen Hawking	British

=>Internal Dtd

```
<!DOCTYPE library [
    <!-- Root Element -->
    <!ELEMENT library (categories+, authors)>

    <!-- Categories -->
    <!ELEMENT categories (name, books)>
    <!ATTLIST categories id ID #REQUIRED>

    <!-- Category Name -->
    <!ELEMENT name (#PCDATA)>


    <!-- Books -->
    <!ELEMENT books (book+)>

    <!-- Book -->
    <!ELEMENT book (title, bookAuthor, publication, price)>
    <!ATTLIST book id ID #REQUIRED>

    <!-- Book Title -->
    <!ELEMENT title (#PCDATA)>

    <!-- Book Author (used inside book) -->
    <!ELEMENT bookAuthor (#PCDATA)>
    <!ATTLIST bookAuthor id IDREF #REQUIRED>

    <!-- Publication -->
    <!ELEMENT publication EMPTY>
    <!ATTLIST publication year CDATA #REQUIRED>

    <!-- Price -->
    <!ELEMENT price (#PCDATA)>
    <!ATTLIST price currency CDATA #REQUIRED>

    <!-- Authors Section -->
    <!ELEMENT authors (author+)>

    <!-- Individual Author -->
```

```
<!ELEMENT author (authorName, nationality)>
<!ATTLIST author id ID #REQUIRED>

<!-- Author Name -->
<!ELEMENT authorName (#PCDATA)>

<!-- Author Nationality -->
<!ELEMENT nationality (#PCDATA)>
]>
```

=>Validating xml



No errors were found

Name : Nisarg Amlani .k.
Roll : CE001
ID : 22ceueg082
Lab : 3

=>Library.xml File

```
<?xml version="1.0" encoding="utf-8" ?>
<library>
  <categories id="cat-1">
    <name>
      Fiction
    </name>
    <books>
      <book id="b-001">
        <title>1984</title>
        <bookAuthor id="A101">George Orwell</bookAuthor>
        <publication year="1949"/>
        <price currency="USD">9.99</price>
      </book>
      <book id="b-002">
        <title>To Kill a Mockingbird</title>
        <bookAuthor id="A102">Harper Lee</bookAuthor>
        <publication year="1960"/>
        <price currency="USD">7.99</price>
      </book>
    </books>
  </categories>
  <categories id="cat-2">
    <name>
      Science
    </name>
    <books>
      <book id="b-003">
        <title>A Brief History of Time</title>
        <bookAuthor id="A103">Stephen Hawking</bookAuthor>
```

```
            <publication year="1988"/>
            <price currency="USD">15.00</price>
        </book>
    </books>
  </categories>
  <authors>
    <author id="A101">
      <authorName>George Orwell</authorName>
      <nationality>British</nationality>
    </author>
    <author id="A102">
      <authorName>Harper Lee</authorName>
      <nationality>American</nationality>
    </author>
    <author id="A103">
      <authorName>Stephen Hawking</authorName>
      <nationality>British</nationality>
    </author>
  </authors>
</library>
```

=>Library.xsd File

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <!-- Define the root element "library" -->
  <xs:element name="library">
    <xs:complexType>
      <xs:sequence>
        <!-- Define the "categories" element -->
        <xs:element name="categories" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="books">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="book" maxOccurs="unbounded">
                      <xs:complexType>
```

```
                        <xs:sequence>
                            <xs:element name="title" type="xs:string"/>
                            <xs:element name="bookAuthor">
                                <xs:complexType>
                                    <xs:simpleContent>
                                        <xs:extension base="xs:string">
                                            <xs:attribute name="id" type="xs:string"
                                                use="required"/>
                                        </xs:extension>
                                    </xs:simpleContent>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="publication">
                                <xs:complexType>
                                    <xs:attribute name="year" type="xs:int" use="required"/>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="price">
                                <xs:complexType>
                                    <xs:simpleContent>
                                        <xs:extension base="xs:decimal">
                                            <xs:attribute name="currency" type="xs:string"
                                                use="required"/>
                                        </xs:extension>
                                    </xs:simpleContent>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:string" use="required"/>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:sequence>
<xs:attribute name="id" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>
<!-- Define the "authors" element -->
<xs:element name="authors">
<xs:complexType>
<xs:sequence>
<xs:element name="author" maxOccurs="unbounded">
<xs:complexType>
```

```
                        <xs:sequence>
                            <xs:element name="authorName" type="xs:string"/>
                            <xs:element name="nationality" type="xs:string"/>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:string" use="required"/>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
</xs:schema>
```

Validation Output :

Name : Nisarg .k. Amlani
Roll : Ce001
Id : 22ceueg082
Lab : 4

=> Soap messages

Request :

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
        <Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://tempuri.org/I
Service1/GetData</Action>
  </s:Header>
  <s:Body>
        <GetData xmlns="http://tempuri.org/">
        <value1>5</value1>
        <value2>9</value2>
        <op>43</op>
        </GetData>
  </s:Body>
</s:Envelope>
```

Response :

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header />
  <s:Body>
        <GetDataResponse xmlns="http://tempuri.org/">
```

```
        <GetDataResult>Your Result is : 14</GetDataResult>
        </GetDataResponse>
  </s:Body>
</s:Envelope>
```

=> WSDL Content

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:wsap="http://schemas.xmlsoap.org/ws/2004/08/addressing/policy"
xmlns:msc="http://schemas.microsoft.com/ws/2005/12/wsdl/contract"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility
-1.0.xsd" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns="http://tempuri.org/" xmlns:wsa10="http://www.w3.org/2005/08/addressing"
xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" name="Service1"
targetNamespace="http://tempuri.org/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
        <xsd:schema targetNamespace="http://tempuri.org/Imports">
        <xsd:import namespace="http://tempuri.org/" />
        <xsd:import namespace="http://schemas.microsoft.com/2003/10/Serialization/" />
        </xsd:schema>
  </wsdl:types>
  <wsdl:message name="IService1_GetData_InputMessage">
        <wsdl:part name="parameters" element="tns:GetData" />
  </wsdl:message>
  <wsdl:message name="IService1_GetData_OutputMessage">
        <wsdl:part name="parameters" element="tns:GetDataResponse" />
  </wsdl:message>
  <wsdl:portType name="IService1">
        <wsdl:operation name="GetData">
        <wsdl:input wsaw:Action="http://tempuri.org/IService1/GetData"
message="tns:IService1_GetData_InputMessage" />
```

```
        <wsdl:output wsaw:Action="http://tempuri.org/IService1/GetDataResponse"
message="tns:IService1_GetData_OutputMessage" />
        </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="BasicHttpBinding_IService1" type="tns:IService1">
        <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
        <wsdl:operation name="GetData">
        <soap:operation soapAction="http://tempuri.org/IService1/GetData"
style="document" />
        <wsdl:input>
        <soap:body use="literal" />
        </wsdl:input>
        <wsdl:output>
        <soap:body use="literal" />
        </wsdl:output>
        </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="Service1">
        <wsdl:port name="BasicHttpBinding_IService1"
binding="tns:BasicHttpBinding_IService1">
        <soap:address
location="http://localhost:8733/Design_Time_Addresses/CalculatorService/Service1/" />
        </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

=> Service Code

```
namespace CalculatorService
{

        public class Service1 : IService1
        {
        public string GetData(int value1 , int value2 , char op)
        {
        double res ;
        switch(op)
```

```
        {
                case '+': res = value1 + value2; break;
                case '-': res = value1 - value2; break;
                case '*': res = value1 * value2; break;
                case '/': res = value1 / value2; break;

                default: return "Invalid Operation";
        }

        return string.Format("Your Result is : {0}",res);
        }


        }
}
```

=> Client Code

```
namespace Forms
{
        public partial class Form1 : Form
        {
        public Form1()
        {
        InitializeComponent();
        }



        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
```

```
        char opr = ' ';
        if (op.Text.Equals("+")) opr = '+';
        if (op.Text.Equals("-")) opr = '-';
        if (op.Text.Equals("*")) opr = '*';
        if (op.Text.Equals("/")) opr = '/';
        ServiceReference2.Service1Client sc = new ServiceReference2.Service1Client();
        label1.Text = sc.GetData(int.Parse(value1.Text), int.Parse(value2.Text), opr);
        }
        }
}
```

=> Service Config File

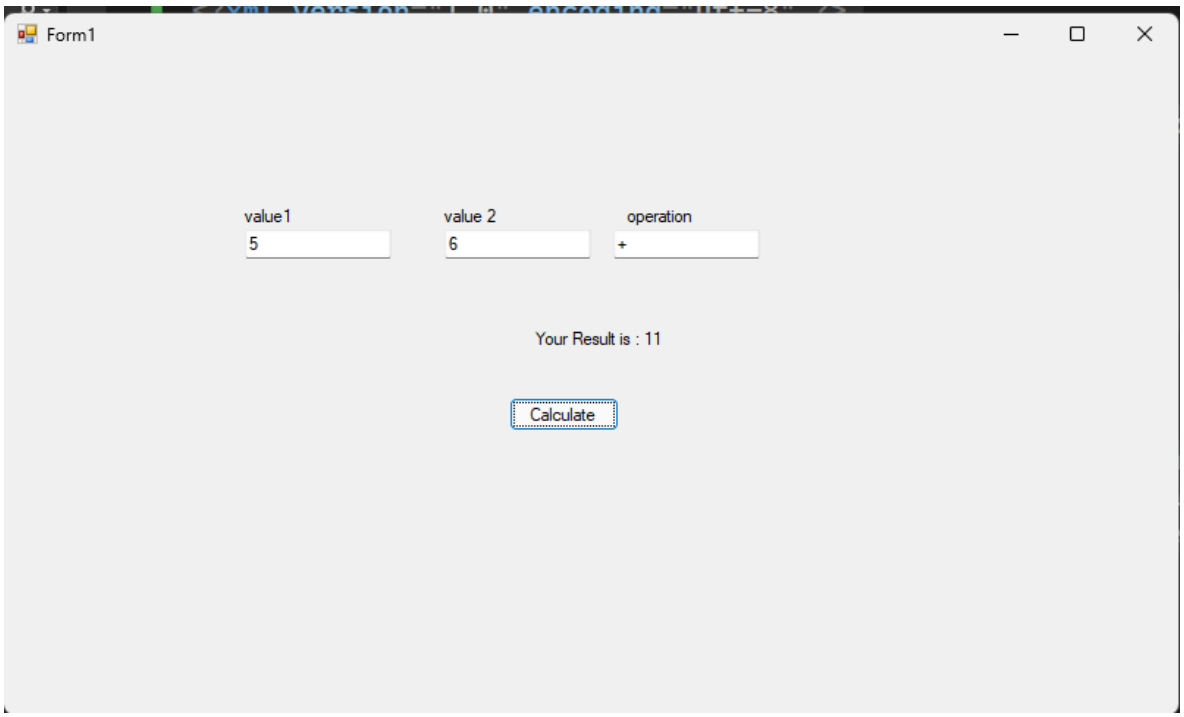
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>

  <appSettings>
        <add key="aspnet:UseTaskFriendlySynchronizationContext" value="true" />
  </appSettings>
  <system.web>
        <compilation debug="true" />
  </system.web>
  <!-- When deploying the service library project, the content of the config file must be
added to the host's
  app.config file. System.Configuration does not support config files for libraries. -->
  <system.serviceModel>
        <services>
        <service name="CalculatorService.Service1">
        <host>
        <baseAddresses>
        <add baseAddress =
"http://localhost:8733/Design_Time_Addresses/CalculatorService/Service1/" />
        </baseAddresses>
        </host>
        <!-- Service Endpoints -->
        <!-- Unless fully qualified, address is relative to base address supplied above -->
```

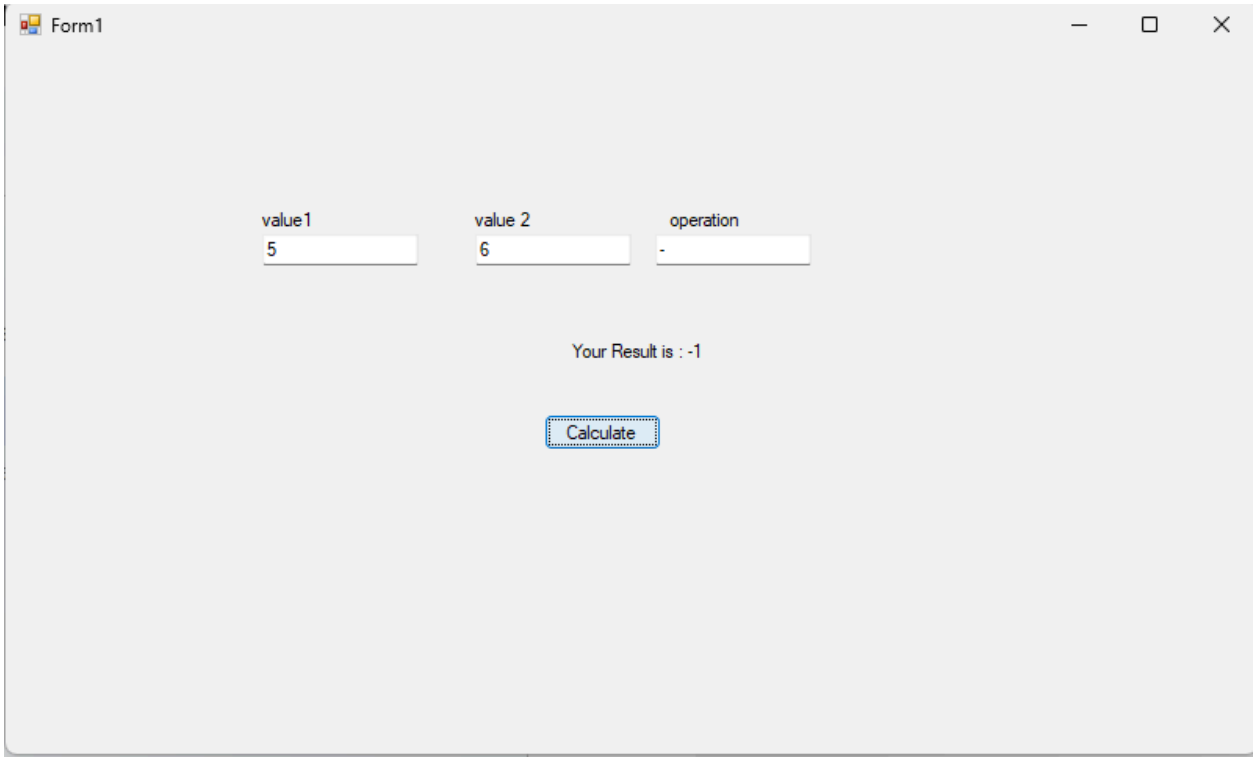
```
        <endpoint address="" binding="basicHttpBinding"
contract="CalculatorService.IService1">
        <!--
        Upon deployment, the following identity element should be removed or replaced
to reflect the
        identity under which the deployed service runs.  If removed, WCF will infer an
appropriate identity
        automatically.
        -->
        <identity>
        <dns value="localhost"/>
        </identity>
        </endpoint>
        <!-- Metadata Endpoints -->
        <!-- The Metadata Exchange endpoint is used by the service to describe itself to
clients. -->
        <!-- This endpoint does not use a secure binding and should be secured or
removed before deployment -->
        <endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange"/>
        </service>
        </services>
        <behaviors>
        <serviceBehaviors>
        <behavior>
        <!-- To avoid disclosing metadata information,
        set the values below to false before deployment -->
        <serviceMetadata httpGetEnabled="True" httpsGetEnabled="True"/>
        <!-- To receive exception details in faults for debugging purposes,
        set the value below to true.  Set to false before deployment
        to avoid disclosing exception information -->
        <serviceDebug includeExceptionDetailInFaults="False" />
        </behavior>
        </serviceBehaviors>
        </behaviors>
  </system.serviceModel>

</configuration>
```

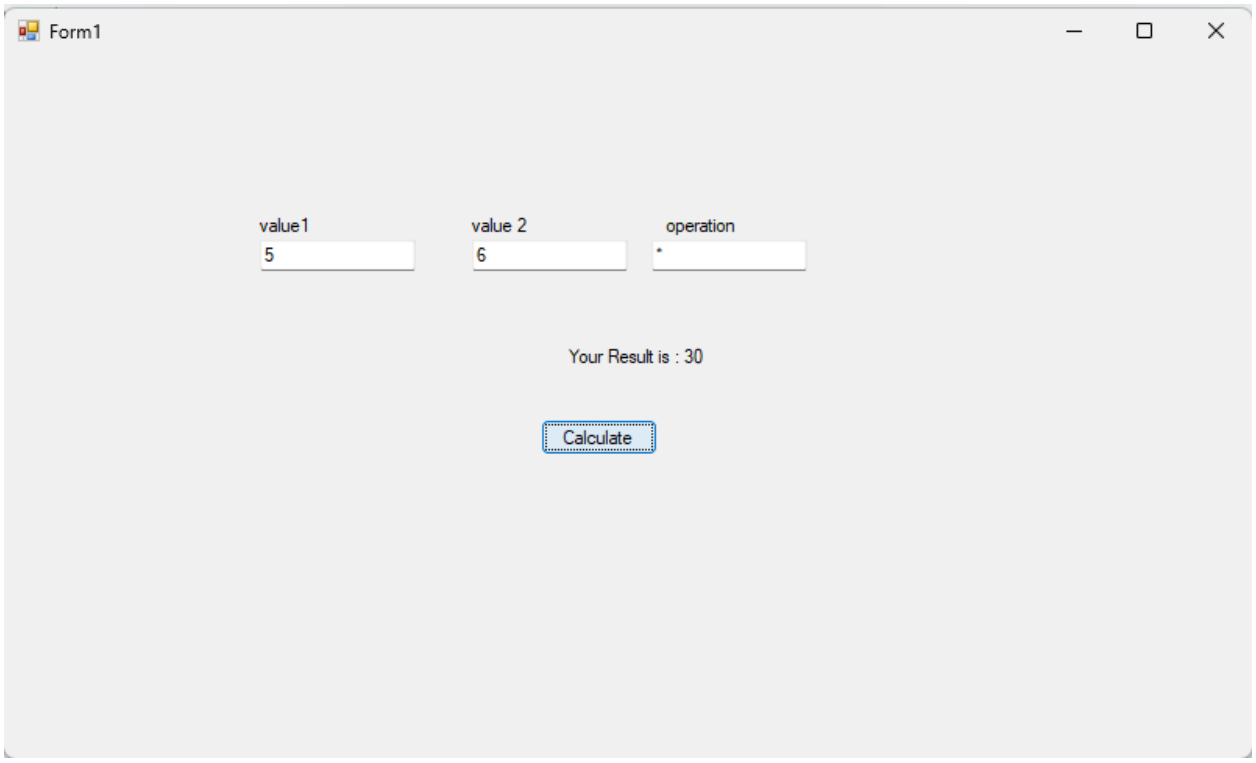
=> Client Config File

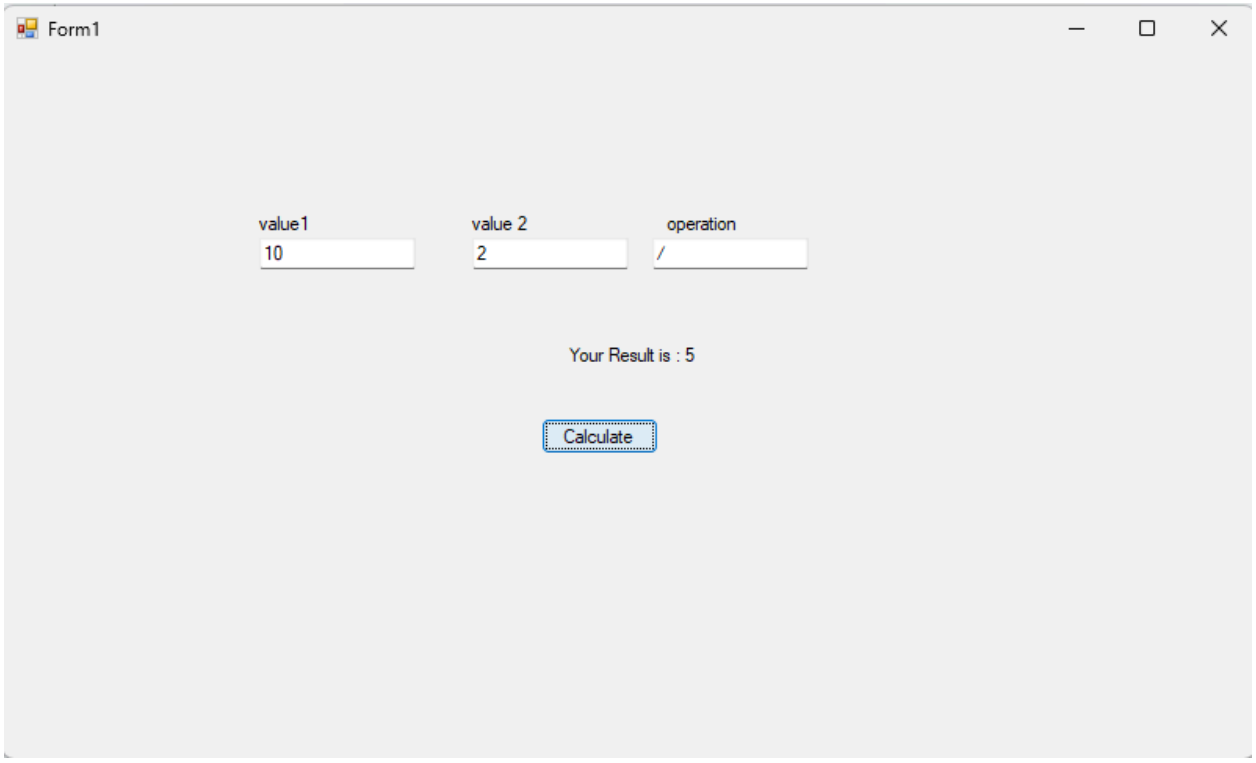
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
        <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
        </startup>
        <system.serviceModel>
        <bindings>
        <basicHttpBinding>
                <binding name="BasicHttpBinding_IService1" />
        </basicHttpBinding>
        </bindings>
        <client>
        <endpoint
address="http://localhost:8733/Design_Time_Addresses/CalculatorService/Service1/"
                binding="basicHttpBinding"
bindingConfiguration="BasicHttpBinding_IService1"
                contract="ServiceReference2.IService1"
name="BasicHttpBinding_IService1" />
        </client>
        </system.serviceModel>
</configuration>
```

=> Outputs


Form1
value1
5
value 2
6
operation
+
Your Result is : 11
Calculate


Form1
value1
5
value 2
6
operation
-
Your Result is : -1
Calculate


Form1
value1
5
value 2
6
operation
*
Your Result is : 30
Calculate


Form1
value1
10
value 2
2
operation
/
Your Result is : 5
Calculate

Name :- Nisarg .k. Amlani
Roll :- CE001
Id :- 22ceueg082
Lab :- 5

---

1. Complex Number Data Contact Code and its config files

=> Host Config file

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

  <appSettings>
        <add key="aspnet:UseTaskFriendlySynchronizationContext" value="true" />
  </appSettings>
  <system.web>
        <compilation debug="true" />
  </system.web>
  <!-- When deploying the service library project, the content of the config file must be
added to the host's
  app.config file. System.Configuration does not support config files for libraries. -->
  <system.serviceModel>
        <services>
        <service name="Lab_5_DataContract.Service1">
        <host>
        <baseAddresses>
        <add baseAddress =
"http://localhost:8733/Design_Time_Addresses/Lab_5_DataContract/Service1/" />
        </baseAddresses>
        </host>
        <!-- Service Endpoints -->
        <!-- Unless fully qualified, address is relative to base address supplied above -->
```

```
        <endpoint address="" binding="basicHttpBinding"
contract="Lab_5_DataContract.IService1">
        <!--
        Upon deployment, the following identity element should be removed or replaced
to reflect the
        identity under which the deployed service runs.  If removed, WCF will infer an
appropriate identity
        automatically.
        -->
        <identity>
        <dns value="localhost"/>
        </identity>
        </endpoint>
        <!-- Metadata Endpoints -->
        <!-- The Metadata Exchange endpoint is used by the service to describe itself to
clients. -->
        <!-- This endpoint does not use a secure binding and should be secured or
removed before deployment -->
        <endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange"/>
        </service>
        </services>
        <behaviors>
        <serviceBehaviors>
        <behavior>
        <!-- To avoid disclosing metadata information,
        set the values below to false before deployment -->
        <serviceMetadata httpGetEnabled="True" httpsGetEnabled="True"/>
        <!-- To receive exception details in faults for debugging purposes,
        set the value below to true.  Set to false before deployment
        to avoid disclosing exception information -->
        <serviceDebug includeExceptionDetailInFaults="False" />
        </behavior>
        </serviceBehaviors>
        </behaviors>
  </system.serviceModel>

</configuration>
```

=> Interface Code

```
namespace Lab_5_DataContract
{
	[ServiceContract]
	internal interface IService1
	{
	[OperationContract]
	ComplexNumber Add(ComplexNumber number1, ComplexNumber number2);

	[OperationContract]
	ComplexNumber Substract (ComplexNumber number1, ComplexNumber
number2);
	}
}
```

=> Interface Implementation

```
namespace Lab_5_DataContract
{
	// NOTE: You can use the "Rename" command on the "Refactor" menu to
change the class name "Service1" in both code and config file together.
	public class Service1 : IService1
	{
	public ComplexNumber Add(ComplexNumber number1, ComplexNumber
number2)
	{
	ComplexNumber result = new ComplexNumber();
	result.Real = number1.Real + number2.Real;
	result.Imaginary = number1.Imaginary + number2.Imaginary;

	return result;

	}

	public ComplexNumber Substract(ComplexNumber number1, ComplexNumber
number2)
	{
	ComplexNumber result = new ComplexNumber();
```

```
        result.Real = number1.Real - number2.Real;
        result.Imaginary = number1.Imaginary - number2.Imaginary;
        return result;
        }
        }
}
```

=> Class Code

```
namespace Lab_5_DataContract
{

        [DataContract]
        public class ComplexNumber
        {
        [DataMember]
        public double Real { get; set; }
        [DataMember]
        public double Imaginary { get; set; }

        }
}
```

=> Client Config File

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
        <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
        </startup>
        <system.serviceModel>
        <bindings>
        <basicHttpBinding>
                <binding name="BasicHttpBinding_IService1" />
        </basicHttpBinding>
        </bindings>
        <client>
```

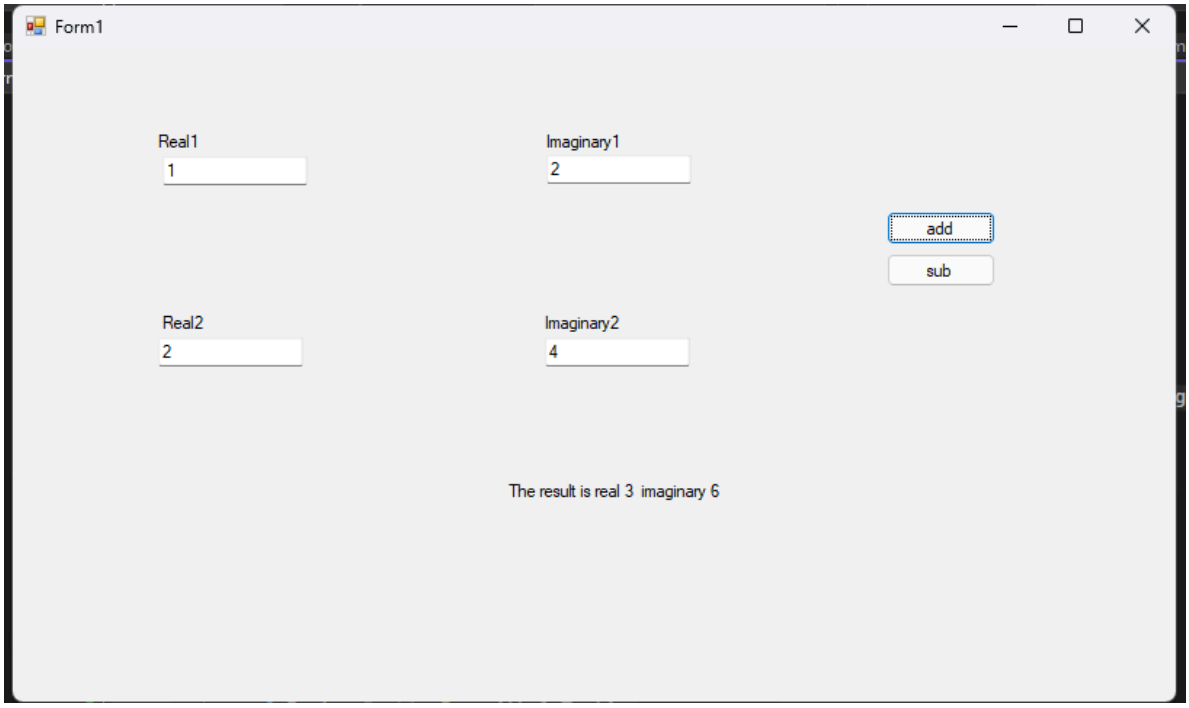
```
        <endpoint
address="http://localhost:8733/Design_Time_Addresses/Lab_5_DataContract/Service1/
"
                binding="basicHttpBinding"
bindingConfiguration="BasicHttpBinding_IService1"
                contract="ServiceReference1.IService1"
name="BasicHttpBinding_IService1" />
        </client>
        </system.serviceModel>
</configuration>
```

=> Client Form Code

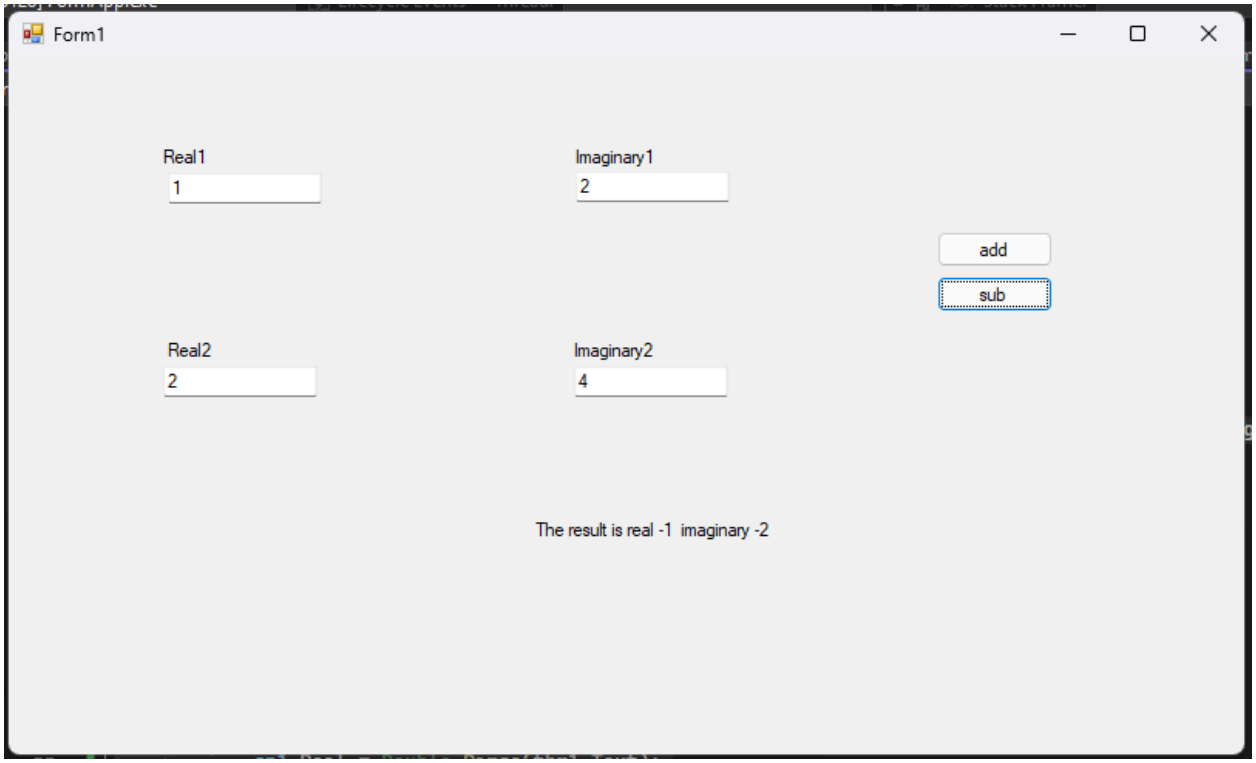
```
namespace FormApp
{
        public partial class Form1 : Form
        {
        public Form1()
        {
        InitializeComponent();
        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void add_Click(object sender, EventArgs e)
        {

        ServiceReference1.Service1Client sc = new ServiceReference1.Service1Client();
        ServiceReference1.ComplexNumber cn1 = new
ServiceReference1.ComplexNumber();
        ServiceReference1.ComplexNumber cn2 = new
ServiceReference1.ComplexNumber();
        ServiceReference1.ComplexNumber res = new
ServiceReference1.ComplexNumber();

        cn1.Real = Double.Parse(tbr1.Text);
        cn1.Imaginary = Double.Parse(tbi1.Text);
```

```
        cn2.Real = Double.Parse(tbr2.Text);
        cn2.Imaginary = Double.Parse(tbi2.Text);
        res = sc.Add(cn1, cn2);




        reslb.Text = string.Format("The result is real {0}  imaginary {1}
",res.Real,res.Imaginary);



        }

        private void sub_Click(object sender, EventArgs e)
        {
        ServiceReference1.Service1Client sc = new ServiceReference1.Service1Client();
        ServiceReference1.ComplexNumber cn1 = new
ServiceReference1.ComplexNumber();
        ServiceReference1.ComplexNumber cn2 = new
ServiceReference1.ComplexNumber();
        ServiceReference1.ComplexNumber res = new
ServiceReference1.ComplexNumber();

        cn1.Real = Double.Parse(tbr1.Text);
        cn1.Imaginary = Double.Parse(tbi1.Text);
        cn2.Real = Double.Parse(tbr2.Text);
        cn2.Imaginary = Double.Parse(tbi2.Text);
        res = sc.Substract(cn1, cn2);

        reslb.Text = string.Format("The result is real {0}  imaginary {1} ", res.Real,
res.Imaginary);



        }
        }
}
```

=> Output Addition



=> Output Substraction



2. WCF Database Demo Service

=> Soap Message

GetAllEmp() :

Request

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
<s:Header>
<Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://tempu
ri.org
/IEmployee/GetAllEmp</Action>
</s:Header>
<s:Body>
<GetAllEmp xmlns="http://tempuri.org/" />
</s:Body>
</s:Envelope>

Response :

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
<s:Header />
<s:Body>
<GetAllEmpResponse xmlns="http://tempuri.org/">
<GetAllEmpResult
xmlns:a="http://schemas.datacontract.org/2004/07/WCF_DatabaseDemo_Servic
e"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
<a:Employee>
<a:Description>Manager</a:Description>
<a:Id>1</a:Id>

<a:Name>Rahul</a:Name>
</a:Employee>
<a:Employee>
<a:Description>HR</a:Description>
<a:Id>2</a:Id>
<a:Name>Rishi</a:Name>
</a:Employee>
<a:Employee>
<a:Description>Tech Lead</a:Description>
<a:Id>3</a:Id>
<a:Name>Pankaj</a:Name>
</a:Employee>
<a:Employee>
<a:Description>CTO</a:Description>
<a:Id>4</a:Id>
<a:Name>Jay</a:Name>
</a:Employee>
</GetAllEmpResult>
</GetAllEmpResponse>
</s:Body>
</s:Envelope>

=> GetEmpById()

Request

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
<s:Header>
<Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://tempu
ri.org
/IEmployee/GetEmpById</Action>
</s:Header>
<s:Body>
<GetEmpById xmlns="http://tempuri.org/">
<id>1</id>

```
</GetEmpById>
</s:Body>
</s:Envelope>
```

Response

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
<s:Header />
<s:Body>
<GetEmpByIdResponse xmlns="http://tempuri.org/">
<GetEmpByIdResult
xmlns:a="http://schemas.datacontract.org/2004/07/WCF_DatabaseDemo_
Service"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
<a:Description>Manager</a:Description>
<a:Id>1</a:Id>
<a:Name>Rahul</a:Name>
</GetEmpByIdResult>
</GetEmpByIdResponse>
</s:Body>
</s:Envelope>
```

=> App.Config Code

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

 <appSettings>
   <add key="aspnet:UseTaskFriendlySynchronizationContext" value="true" />
 </appSettings>
 <system.web>
   <compilation debug="true" />
 </system.web>
 <!-- When deploying the service library project, the content of the config file must
be added to the host's
```

```
 app.config file. System.Configuration does not support config files for libraries.
-->
 <system.serviceModel>
   <services>
     <service name="Data_Contract.EmpService">
       <host>
         <baseAddresses>
           <add baseAddress =
"http://localhost:8733/Design_Time_Addresses/Data_Contract/EmpService/" />
         </baseAddresses>
       </host>
       <!-- Service Endpoints -->
       <!-- Unless fully qualified, address is relative to base address supplied above
-->
       <endpoint address="" binding="basicHttpBinding"
contract="Data_Contract.IEmployee">
         <!--
             Upon deployment, the following identity element should be removed or
replaced to reflect the
             identity under which the deployed service runs.  If removed, WCF will
infer an appropriate identity
             automatically.
         -->
         <identity>
           <dns value="localhost"/>
         </identity>
       </endpoint>
       <!-- Metadata Endpoints -->
       <!-- The Metadata Exchange endpoint is used by the service to describe
itself to clients. -->
       <!-- This endpoint does not use a secure binding and should be secured or
removed before deployment -->
       <endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange"/>
     </service>
   </services>
   <behaviors>
     <serviceBehaviors>
       <behavior>
         <!-- To avoid disclosing metadata information,
```

```
        set the values below to false before deployment -->
        <serviceMetadata httpGetEnabled="True" httpsGetEnabled="True"/>
        <!-- To receive exception details in faults for debugging purposes,
        set the value below to true.  Set to false before deployment
        to avoid disclosing exception information -->
        <serviceDebug includeExceptionDetailInFaults="False" />
      </behavior>
    </serviceBehaviors>
  </behaviors>
 </system.serviceModel>
</configuration>
```

=> Service Interface

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;

namespace Data_Contract
{
  [ServiceContract]
  public interface IEmployee
  {
     [OperationContract]
     DataSet GetAllEmp();
     [OperationContract]
     Employee GetEmpbyID(int id);
  }
  [DataContract]
  public class Employee
  {
     [DataMember]
     public int Id { get; set; }
     [DataMember] public string Name { get; set; }
```

```
    [DataMember] public string Designation { get; set; }
  }
}
```

=> Service Class

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;

namespace Data_Contract
{
  // NOTE: You can use the "Rename" command on the "Refactor" menu to
change the class name "Service1" in both code and config file together.
  public class EmpService : IEmployee
  {
    private string connectionString = "Data Source=(localdb)\\mssqllocaldb;Initial
Catalog=empdb2;Integrated Security=True;Pooling=False;";

      DataSet IEmployee.GetAllEmp()
    {
      DataSet dataSet = new DataSet();
      string query = "SELECT * from emp_info";
      using(SqlConnection connection = new SqlConnection(connectionString))
      {
        SqlDataAdapter dataAdapter = new SqlDataAdapter(query,
connection);
        connection.Open();
        dataAdapter.Fill(dataSet, "Employees");
      }

      return dataSet;
    }
```

```
        Employee IEmployee.GetEmpbyID(int id)
        {
            SqlConnection connection = new SqlConnection(connectionString);
            string query = "select * from emp_info where Id=@EmployeeID";
            SqlDataAdapter dataAdapter = new SqlDataAdapter(query, connection);
            dataAdapter.SelectCommand.Parameters.AddWithValue("@EmployeeID",
id);
            DataSet dataset = new DataSet();
            connection.Open();
            dataAdapter.Fill(dataset, "Employee");
            var row = dataset.Tables["Employee"].Rows[0];
            Employee emp = new Employee();
            emp.Id = Convert.ToInt32(row["Id"]);
            emp.Name = row["Name"].ToString();
            emp.Designation = row["Designation"].ToString();
            return emp;
        }
    }
}
```

Client Code

=> Client App.Config Code

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 For more information on how to configure your ASP.NET application, please visit
 https://go.microsoft.com/fwlink/?LinkId=169433
 -->
<configuration>
 <system.web>
   <compilation debug="true" targetFramework="4.7.2" />
   <httpRuntime targetFramework="4.7.2" />
 </system.web>
 <system.codedom>
   <compilers>
     <compiler language="c#;cs;csharp" extension=".cs"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvi
der, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0,
```

```
Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4"
compilerOptions="/langversion:default /nowarn:1659;1699;1701" />
     <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0,
Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4"
compilerOptions="/langversion:default /nowarn:41008
/define:_MYTYPE=\&quot;Web\&quot; /optionInfer+" />
   </compilers>
 </system.codedom>
 <system.serviceModel>
   <bindings>
     <basicHttpBinding>
       <binding name="BasicHttpBinding_IEmployee" />
     </basicHttpBinding>
   </bindings>
   <client>
     <endpoint
address="http://localhost:8733/Design_Time_Addresses/Data_Contract/EmpServ
ice/"
       binding="basicHttpBinding"
bindingConfiguration="BasicHttpBinding_IEmployee"
       contract="ServiceReference1.IEmployee"
name="BasicHttpBinding_IEmployee" />
   </client>
 </system.serviceModel>
</configuration>
```

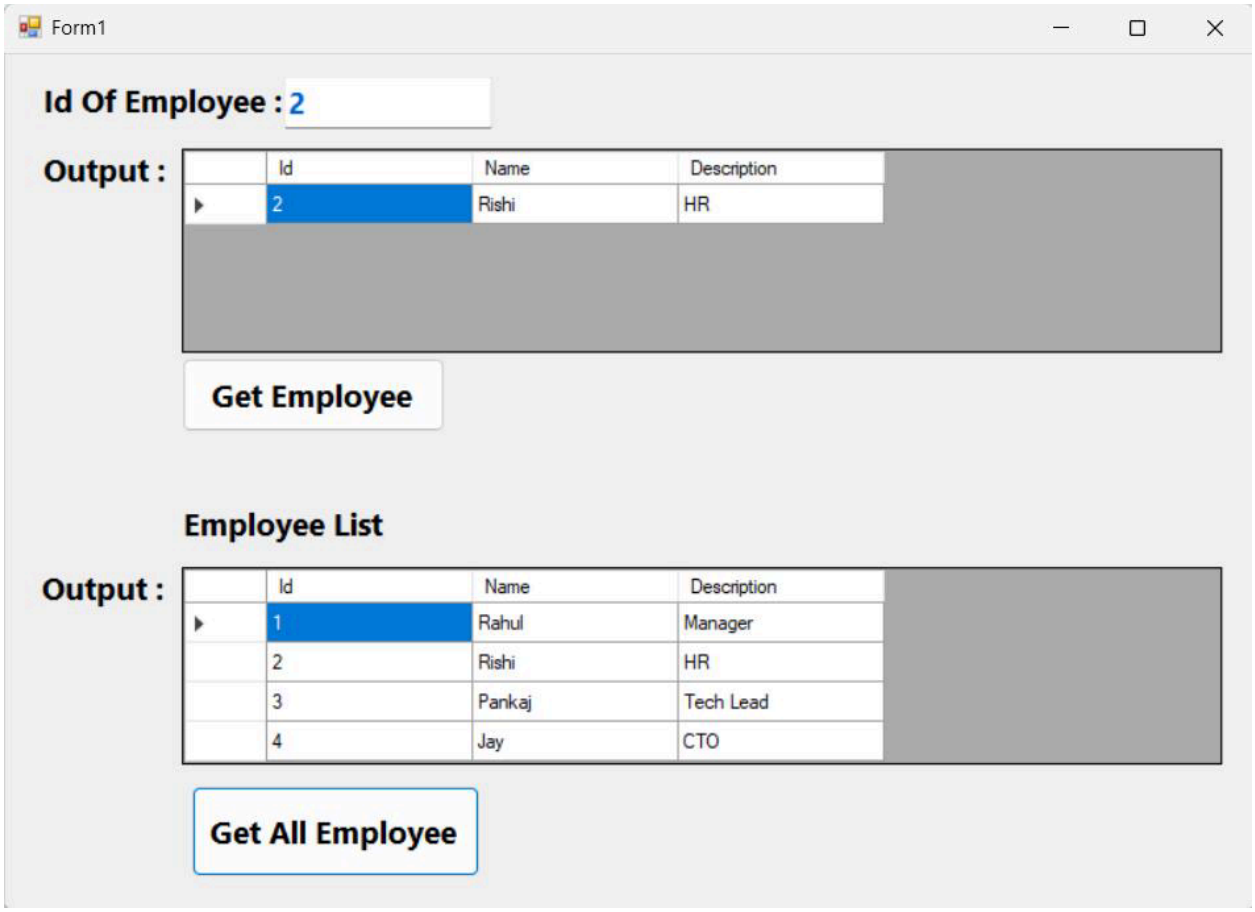
=> Client Form Code

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

```
namespace Employee_client
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
        }

        protected void GetAll_Click(object sender, EventArgs e)
        {
            ServiceReference1.EmployeeClient employeeClient =
                new ServiceReference1.EmployeeClient();
            DataSet ds = new DataSet();
            ds = employeeClient.GetAllEmp();
            GridView1.DataSource = ds.Tables[0];

            GridView1.DataBind();
        }

        protected void GetNameByID_Click(object sender, EventArgs e)
        {
            ServiceReference1.EmployeeClient employeeClient =
                new ServiceReference1.EmployeeClient();

            ServiceReference1.EmployeeClient employeeClient1 = new
ServiceReference1.EmployeeClient();

            ServiceReference1.Employee emp =
employeeClient1.GetEmpbyID(Int32.Parse(TB.Text));

            lbl.Text = emp.Name;
        }
    }
}
```

=> webform.aspx code

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs" Inherits="Employee_client.WebForm1" %>

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <asp:Button ID="GetAll" runat="server" OnClick="GetAll_Click" Text="Get
All" />
      <br />
      <br />
      <br />
      <asp:GridView ID="GridView1" runat="server">
      </asp:GridView>
      <br />
      <br />
      <asp:TextBox ID="TB" runat="server"></asp:TextBox>
      <br />
      <br />
      <asp:Button ID="GetNameByID" runat="server"
OnClick="GetNameByID_Click" Text="GetNameByid" />
      <br />
      <br />
      <asp:Label ID="lbl" runat="server" Text="Name : label"></asp:Label>
      <br />
    </div>
  </form>
</body>
</html>
```


Form1
Id Of Employee : 2
Output :
Id
Name
Description
2
Rishi
HR
Get Employee
Employee List
Output :
Id
Name
Description
1
Rahul
Manager
2
Rishi
HR
3
Pankaj
Tech Lead
4
Jay
CTO
Get All Employee

Lab :- 6
Name :- Nisarg .K. Amlani
Roll :- Ce001
Id :- 22ceueg082

---

1) Use the calculator WCF service/ any other WCF service developed in the previous labs.

2) Host the service in the Windows Form application. Define metadata and application endpoints over TCP protocol in the host application.

3) Develop a Windows Form client application to consume the service.

4) Update the host and client applications to provide options for consuming the service with different protocols (Pipe, HTTP and TCP).

Code IService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.ServiceModel;
using System.Text;
using System.Threading.Tasks;

namespace WCFServiceLib
{
    [ServiceContract]
    public interface IService
    {
        [OperationContract]
        string GetData(int value1, int value2, char op);
    }
}
```

Code Service.cs

```
namespace WCFServiceLib
{
    public class Service : IService
    {
        public string GetData(int value1, int value2, char op)
        {
            double res ;
            switch(op)

            {
                case '+': res = value1 + value2; break;
                case '-': res = value1 - value2; break;
                case '*': res = value1 * value2; break;
                case '/': res = value1 / value2; break;
                default: return "Invalid Operation";
            }
            return string.Format("Your Result is : {0}",res);

        }
    }
}
```

Code Form1.cs (Host App)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.ServiceModel;
using System.ServiceModel.Description;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using WCFServiceLib;

namespace WcfServiceHost
{

    public partial class Form1 : Form
    {
        private ServiceHost _sh = null;
        public Form1()
        {
            InitializeComponent();
        }

        protected override void OnLoad(EventArgs e)
        {
            base.OnLoad(e);
            Uri tcp = new Uri("net.tcp://localhost:8010/TcpBinding");
            Uri pipe = new Uri("net.pipe://localhost/NetNamedPipeBinding");
            Uri http = new Uri("http://localhost:8733/Design_Time_Addresses");

            _sh = new ServiceHost(typeof(Service), tcp,http,pipe);
            NetTcpBinding tcpb = new NetTcpBinding();
            NetNamedPipeBinding nppb = new NetNamedPipeBinding();
            NetHttpBinding httpb = new NetHttpBinding();
            ServiceMetadataBehavior metadataBehavior = new ServiceMetadataBehavior();
            _sh.Description.Behaviors.Add(metadataBehavior);
```

```
_sh.AddServiceEndpoint(typeof(IMetadataExchange),MetadataExchangeBindings.Crea
teMexTcpBinding(), "mex");
            _sh.AddServiceEndpoint(typeof(IService), tcpb, tcp);

            _sh.AddServiceEndpoint(typeof(IMetadataExchange),
MetadataExchangeBindings.CreateMexNamedPipeBinding(),
                "mex");
            _sh.AddServiceEndpoint(typeof(IService), nppb, pipe);

            _sh.AddServiceEndpoint(typeof(IMetadataExchange),
MetadataExchangeBindings.CreateMexHttpBinding(), "mex");
            _sh.AddServiceEndpoint(typeof(IService), httpb, http);

            _sh.Open();
            reslbl.Text = "Service is running...";
        }
    }
}
```

Code Form1.cs(Client App)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using WCFClient.Service;

namespace WCFClient
{
    public partial class Form1 : Form
    {
        public Form1()
        {
```

```
            InitializeComponent();
        }

        private void Calculate_Click(object sender, EventArgs e)
        {
            char operation = ' ';
            if (opr.Text.Equals("+")) operation = '+';
            if (opr.Text.Equals("-")) operation = '-';
            if (opr.Text.Equals("*")) operation = '*';
            if (opr.Text.Equals("/")) operation = '/';

            WCFClient.Service.ServiceClient client ;
            WCFClient.Service.ServiceClient client1 =
                new WCFClient.Service.ServiceClient(

WCFClient.Service.ServiceClient.EndpointConfiguration.NetHttpBinding_IService);
            WCFClient.Service.ServiceClient client2 =
                new WCFClient.Service.ServiceClient(

WCFClient.Service.ServiceClient.EndpointConfiguration.NetTcpBinding_IService);

            client = client1;
            // client = client2;

            string res = client.GetData(int.Parse(value1.Text), int.Parse(value2.Text),
operation);
            this.result.Text = res;
        }
    }
}
```

Code App.config (Service)

```
<?xml version="1.0" encoding="utf-8" ?>
<confi guration>
<appSettings>
<add key="aspnet:UseTaskFriendlySynchronizationContext" value="true" />
</appSettings>
<system.web>
<compilation debug="true" />
```

```
</system.web>
<!-- When deploying the service library project, the content of the confi g fi le must be added to the host's
app.confi g fi le. System.Confi guration does not support confi g fi les for libraries. -->
<system.serviceModel>
<services>
<service name="WCFServiceLib.Service">
<host>
<baseAddresses>
<add baseAddress =
"http://localhost:8733/Design_Time_Addresses/WCFServiceLib/Service/" />
</baseAddresses>
</host>
<!-- Service Endpoints -->
<!-- Unless fully qualifi ed, address is relative to base address supplied above -->
<endpoint address="" binding="basicHttpBinding" contract="WCFServiceLib.IService">
<!--
Upon deployment, the following identity element should be removed or replaced to refl ect the
identity under which the deployed service runs. If removed, WCF will infer an appropriate identity
automatically.
-->
<identity>
<dns value="localhost"/>
</identity>
</endpoint>
<!-- Metadata Endpoints -->
<!-- The Metadata Exchange endpoint is used by the service to describe itself to clients.
→

<!-- This endpoint does not use a secure binding and should be secured or removed before deployment -->
<endpoint address="mex" binding="mexHttpBinding" contract="IMetadataExchange"/>
</service>
</services>
<behaviors>
<serviceBehaviors>
<behavior>
<!-- To avoid disclosing metadata information,
```

```
set the values below to false before deployment -->
<serviceMetadata httpGetEnabled="True" httpsGetEnabled="True"/>
<!-- To receive exception details in faults for debugging purposes,
set the value below to true. Set to false before deployment
to avoid disclosing exception information -->
<serviceDebug includeExceptionDetailInFaults="False" />
</behavior>
</serviceBehaviors>
</behaviors>
</system.serviceModel>
</confi guration>
```

Code App.config (Host App)

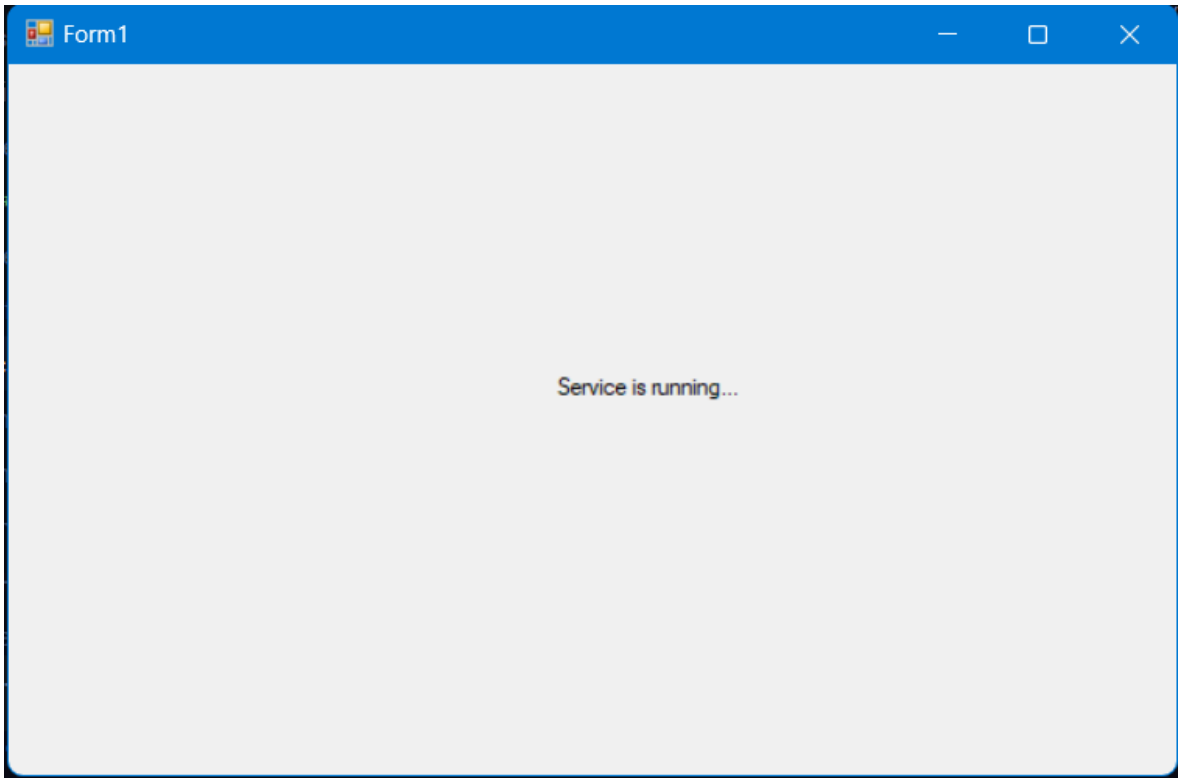
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
   <startup>
      <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.8" />
   </startup>
</configuration>
```

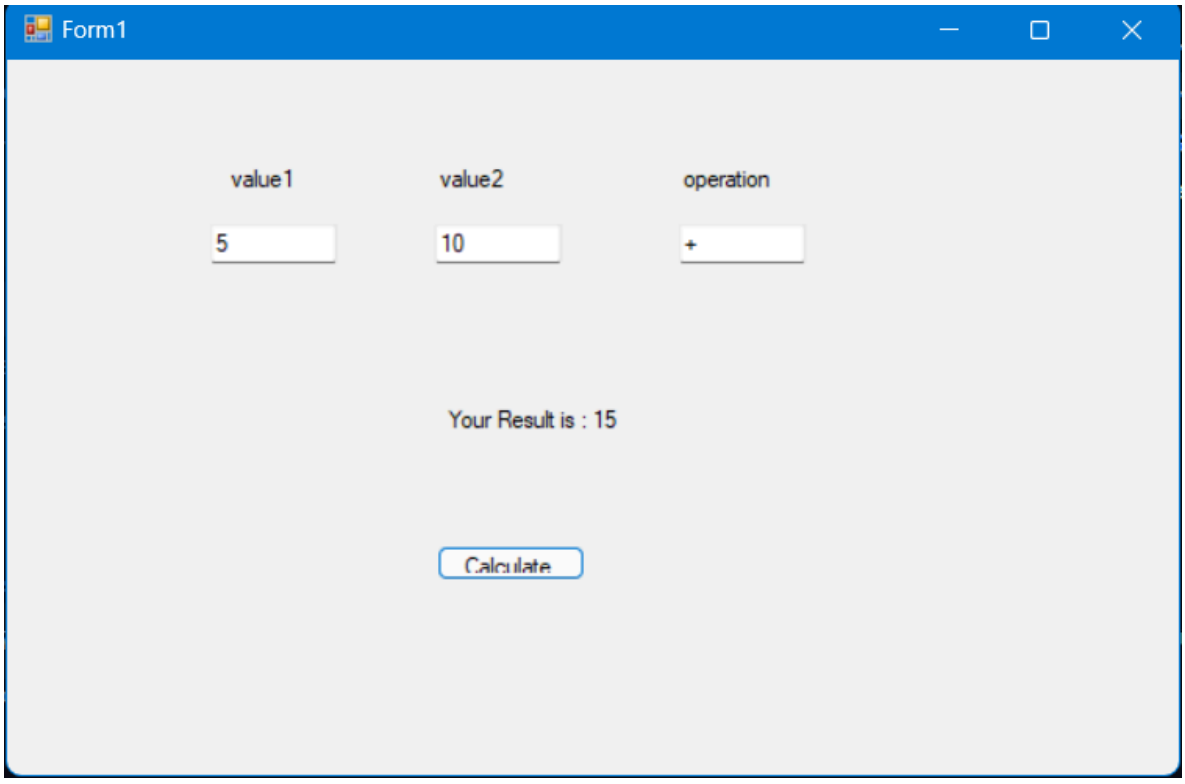
Code App.config (Client App)

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
   <startup>
      <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.8" />
   </startup>
   <system.serviceModel>
      <bindings>
         <netTcpBinding>
                              <binding name="NetTcpBinding_IService">
                                    <security>
                                          <transport sslProtocols="None" />
                                    </security>
                              </binding>
         </netTcpBinding>
```

```
      </bindings>
      <client>
         <endpoint address="net.tcp//localhost:8010/TcpBinding"
binding="netTcpBinding"
         bindingConfiguration="NetTcpBinding_IService" contract="tcp.IService"
name="NetTcpBinding_IService">
            <identity>
               <userPrincipalName value="NISARG\CEDDU" />
            </identity>
         </endpoint>
         <endpoint address="net.pipe://localhost/NetNamedPipeBinding"
                  binding="netNamedPipeBinding"
bindingConfiguration="NetNamedPipeBinding_IService"
                  contract="http.IService" name="NetNamedPipeBinding_IService">
            <identity>
               <userPrincipalName value="Nisarg" />
            </identity>
         </endpoint>
         <endpoint address="http://localhost:8733/Design_Time_Addresses"
                  binding="customBinding" bindingConfiguration="NetHttpBinding_IService"
                  contract="http.IService" name="NetHttpBinding_IService" />
      </client>
   </system.serviceModel>
</configuration>
```

Output :


Form1
Service is running...


Form1
value1
value2
operation
5
10
+
Your Result is : 15
Calculate

Name : Nisarg .k. Amlani
Roll : Ce001
Lab : 07
Id : 22ceueg082

WCF Message Contract Code (Service)

=> App.Config

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


 <appSettings>
   <add key="aspnet:UseTaskFriendlySynchronizationContext" value="true" />
 </appSettings>
 <system.web>
   <compilation debug="true" />
 </system.web>
 <!-- When deploying the service library project, the content of the config file must be
added to the host's
 app.config file. System.Configuration does not support config files for libraries. -->
 <system.serviceModel>
   <services>
     <service name="Message_Contract.Service1">
       <host>
         <baseAddresses>
           <add baseAddress =
"http://localhost:8733/Design_Time_Addresses/Message_Contract/Service1/" />
         </baseAddresses>
       </host>
       <!-- Service Endpoints -->
```

```
      <!-- Unless fully qualified, address is relative to base address supplied above -->
      <endpoint address="" binding="basicHttpBinding"
contract="Message_Contract.IService1">
        <!--
            Upon deployment, the following identity element should be removed or replaced
to reflect the
            identity under which the deployed service runs.  If removed, WCF will infer an
appropriate identity
            automatically.
        -->
        <identity>
          <dns value="localhost"/>
        </identity>
      </endpoint>
      <!-- Metadata Endpoints -->
      <!-- The Metadata Exchange endpoint is used by the service to describe itself to
clients. -->
      <!-- This endpoint does not use a secure binding and should be secured or
removed before deployment -->
      <endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange"/>
    </service>
  </services>
  <behaviors>
    <serviceBehaviors>
      <behavior>
        <!-- To avoid disclosing metadata information,
        set the values below to false before deployment -->
        <serviceMetadata httpGetEnabled="True" httpsGetEnabled="True"/>
        <!-- To receive exception details in faults for debugging purposes,
        set the value below to true.  Set to false before deployment
        to avoid disclosing exception information -->
        <serviceDebug includeExceptionDetailInFaults="False" />
      </behavior>
    </serviceBehaviors>
  </behaviors>
 </system.serviceModel>


</configuration>
```

=> IService.cs Code

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;


namespace Message_Contract
{
  // NOTE: You can use the "Rename" command on the "Refactor" menu to change the interface name "IService1" in both code and config file together.
  [ServiceContract]
  public interface IService1
  {
     [OperationContract]
     string InitiateOrder();


     [OperationContract]
     BookOrder PlaceOrder(BookOrder request);


     [OperationContract]
     string FinalizeOrder();


     // TODO: Add your service operations here
  }


  [MessageContract]
  public class BookOrder
  {
     private string isbn;
     private int quantity;
     private string firstname;
     private string lastname;
```

```
private string address;
private string ordernumber;


public BookOrder()
{
}


public BookOrder(BookOrder message)
{
    this.isbn = message.isbn;
    this.quantity = message.quantity;
    this.firstname = message.firstname;
    this.lastname = message.lastname;
    this.address = message.address;
    this.ordernumber = message.ordernumber;
}


[MessageHeader]
public string ISBN
{
    get { return isbn; }
    set { isbn = value; }
}


[MessageBodyMember]
public int Quantity
{
    get { return quantity; }
    set { quantity = value; }
}

[MessageBodyMember]
public string Firstname
{
    get { return firstname; }
    set { firstname = value; }
```

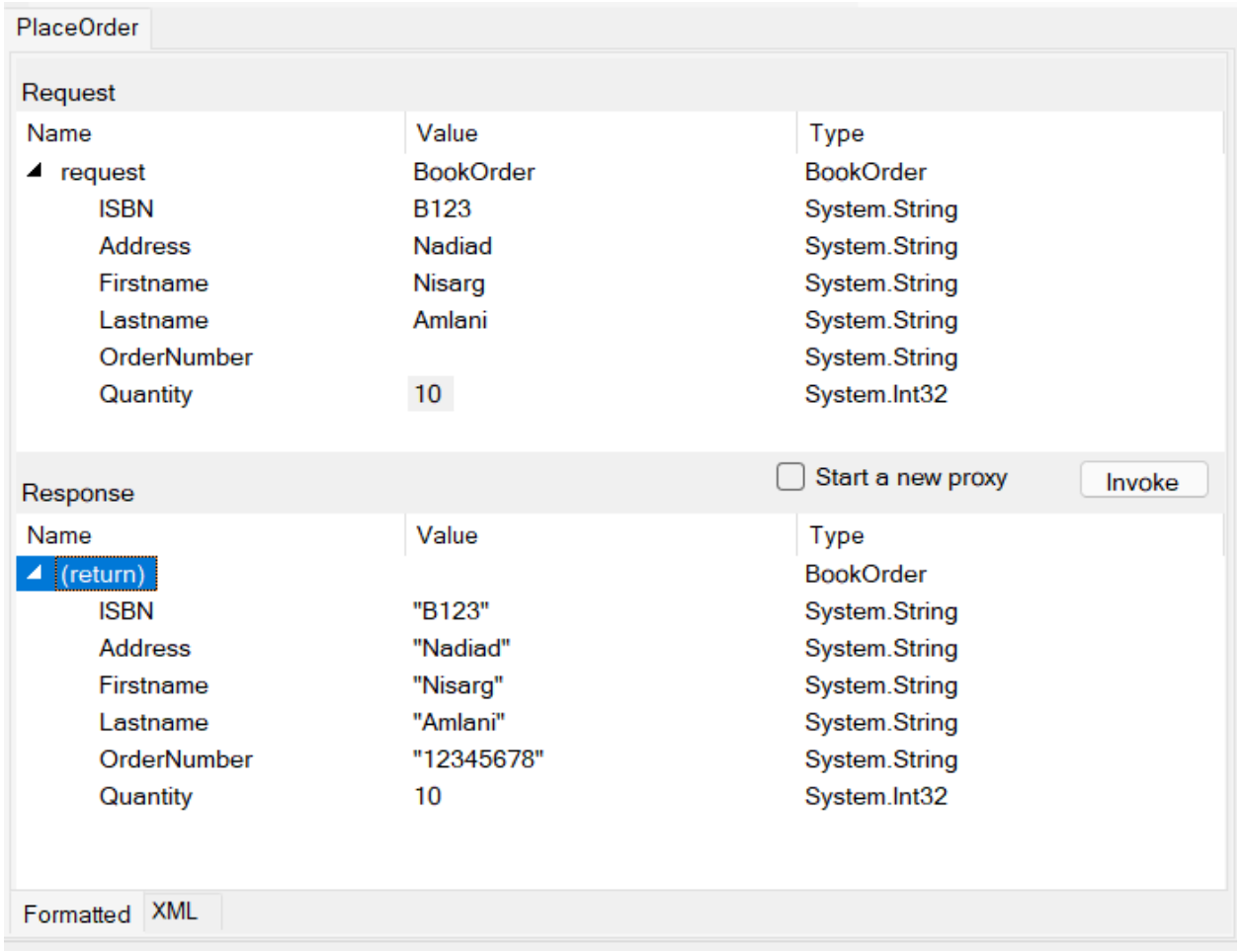
```
        }


        [MessageBodyMember]
        public string Lastname
        {
            get { return lastname; }
            set { lastname = value; }
        }


        [MessageBodyMember]
        public string Address
        {
            get { return address; }
            set { address = value; }
        }


        [MessageBodyMember]
        public string OrderNumber
        {
            get { return ordernumber; }
            set { ordernumber = value; }
        }
    }
}
```

=> Service.cs Code

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;


namespace Message_Contract
```

```
{
   // NOTE: You can use the "Rename" command on the "Refactor" menu to change the
class name "Service1" in both code and config file together.
   public class Service1 : IService1
   {
      string IService1.InitiateOrder()
      {
         return "Initiating Order...";
      }


       BookOrder IService1.PlaceOrder(BookOrder request)
      {
         BookOrder response = new BookOrder(request);
         response.OrderNumber = "12345678";
         return response;
      }


      string IService1.FinalizeOrder()
      {
         return "Order placed sucessfully.";
      }
   }
}
```

=> Output (Service)



PlaceOrder

Request

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
 <s:Header>
  <Action s:mustUnderstand="1" xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">
  <h:ISBN xmlns:h="http://tempuri.org/">B123</h:ISBN>
 </s:Header>
 <s:Body>
  <BookOrder xmlns="http://tempuri.org/">
   <Address>Nadiad</Address>
   <Firstname>Nisarg</Firstname>
   <Lastname>Amlani</Lastname>
   <OrderNumber />
   <Quantity>10</Quantity>
  </BookOrder>
 </s:Body>
```

Response

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
 <s:Header>
  <h:ISBN xmlns:h="http://tempuri.org/">B123</h:ISBN>
 </s:Header>
 <s:Body>
  <BookOrder xmlns="http://tempuri.org/">
   <Address>Nadiad</Address>
   <Firstname>Nisarg</Firstname>
   <Lastname>Amlani</Lastname>
   <OrderNumber>12345678</OrderNumber>
   <Quantity>10</Quantity>
  </BookOrder>
 </s:Body>
</s:Envelope>
```

=> Windows Form App (Host)

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <startup>
     <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
  <system.serviceModel>

 <services>
       <service name="Message_Contract.Service1"
behaviorConfiguration="metadataSupport">
         <host>
           <baseAddresses>
             <add baseAddress="http://localhost:8733/Design_Time_Addresses/" />
             <add baseAddress="net.pipe://localhost/Message_Contract" />
             <add baseAddress="net.tcp://localhost:8000/Message_Contract" />
           </baseAddresses>
         </host>


         <endpoint address="" binding="wsHttpBinding"
contract="Message_Contract.IService1" />


         <endpoint address="tcpmex" binding="mexTcpBinding"
contract="IMetadataExchange" />
         <endpoint address="namedpipemex" binding="mexNamedPipeBinding"
contract="IMetadataExchange" />


       </service>
     </services>
     <behaviors>
       <serviceBehaviors>
         <behavior name="metadataSupport">
           <serviceMetadata httpGetEnabled="false" httpGetUrl="" />
         </behavior>
       </serviceBehaviors>
     </behaviors>
```

```
  </system.serviceModel>
</configuration>
```

=> App.config (host)

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <startup>
     <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
  <system.serviceModel>

 <services>
        <service name="Message_Contract.Service1"
behaviorConfiguration="metadataSupport">
          <host>
            <baseAddresses>
              <add baseAddress="http://localhost:8733/Design_Time_Addresses/" />
              <add baseAddress="net.pipe://localhost/Message_Contract" />
              <add baseAddress="net.tcp://localhost:8000/Message_Contract" />
            </baseAddresses>
          </host>


          <endpoint address="" binding="wsHttpBinding"
contract="Message_Contract.IService1" />


          <endpoint address="tcpmex" binding="mexTcpBinding"
contract="IMetadataExchange" />
          <endpoint address="namedpipemex" binding="mexNamedPipeBinding"
contract="IMetadataExchange" />


        </service>
      </services>
      <behaviors>
        <serviceBehaviors>
```

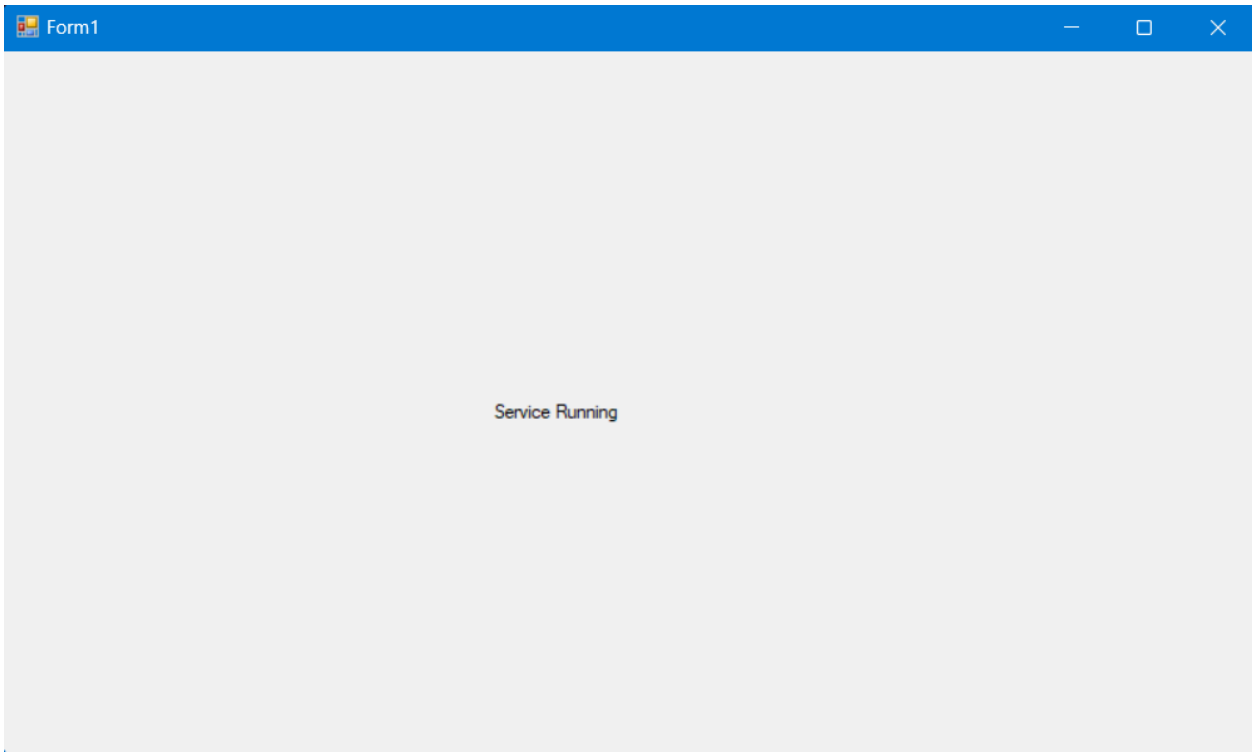
```
        <behavior name="metadataSupport">
          <serviceMetadata httpGetEnabled="false" httpGetUrl="" />
        </behavior>
      </serviceBehaviors>
    </behaviors>
  </system.serviceModel>
</configuration>
```

=> Form.cs (Host)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.ServiceModel;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;


namespace WindowsFormsAppHost
{
  public partial class Form1 : Form
  {
    ServiceHost sh = null;
    public Form1()
    {
      InitializeComponent();
    }

    private void Form1_FormClosing(object sender ,
FormClosedEventArgs e)
    {
      sh.Close();
    }
```

```
        private void Form1_Load_1(object sender, EventArgs e)
        {
            sh = new ServiceHost(typeof(Message_Contract.Service1));
            sh.Open();
            lbl1.Text = "Service Running";
        }
    }
}
```

=> Output (Host)



=> Windows Form App (Client)

=> App.Config Code (Client)

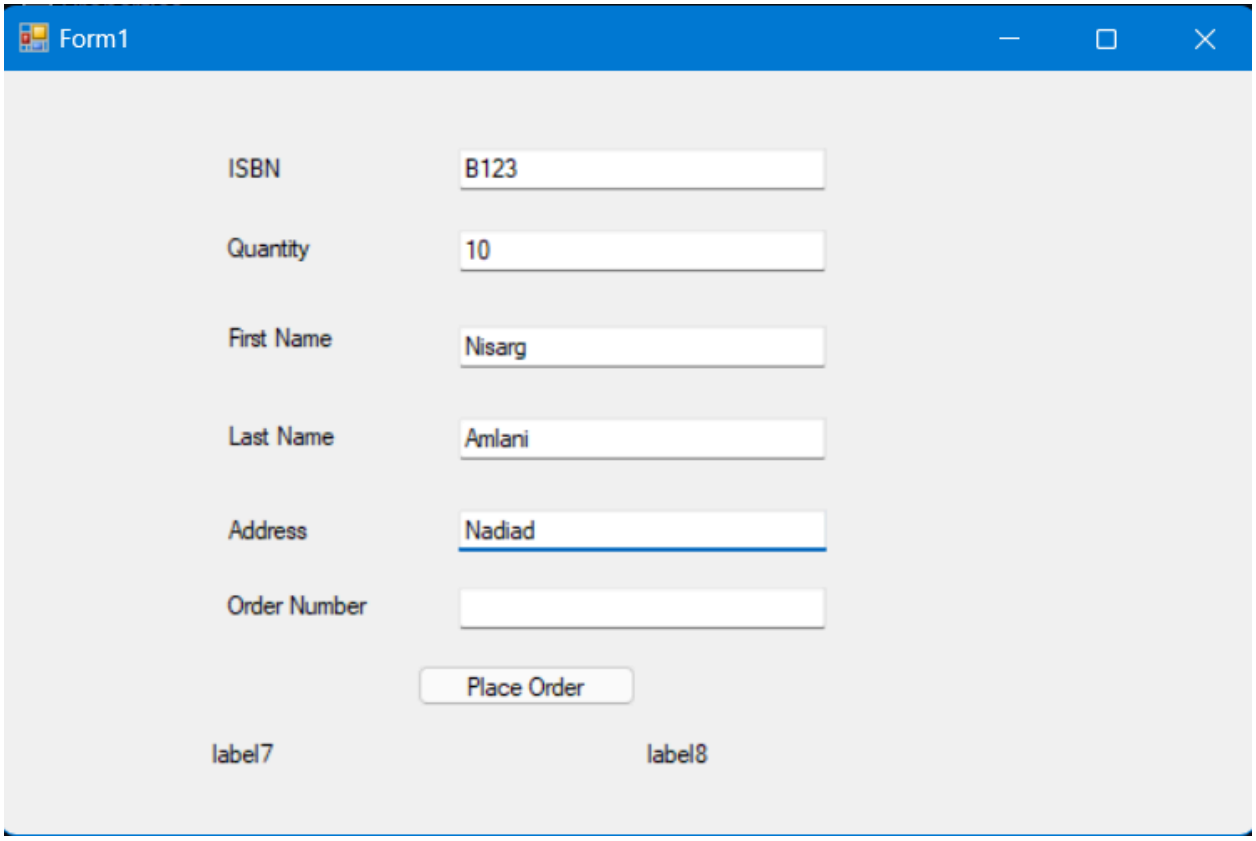
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.8.1" />
  </startup>
  <system.serviceModel>
    <bindings>
      <wsHttpBinding>
        <binding name="WSHttpBinding_IService" />
        <binding name="WSHttpBinding_IService1" />
      </wsHttpBinding>
    </bindings>
    <client>


      <endpoint address="http://localhost:8733/Design_Time_Addresses/"
        binding="wsHttpBinding" bindingConfiguration="WSHttpBinding_IService1"
        contract="TCP.IService1" name="WSHttpBinding_IService1">
        <identity>
          <userPrincipalName value="NISARGSLIG3\LENOVO" />
        </identity>
      </endpoint>
    </client>
  </system.serviceModel>
</configuration>
```

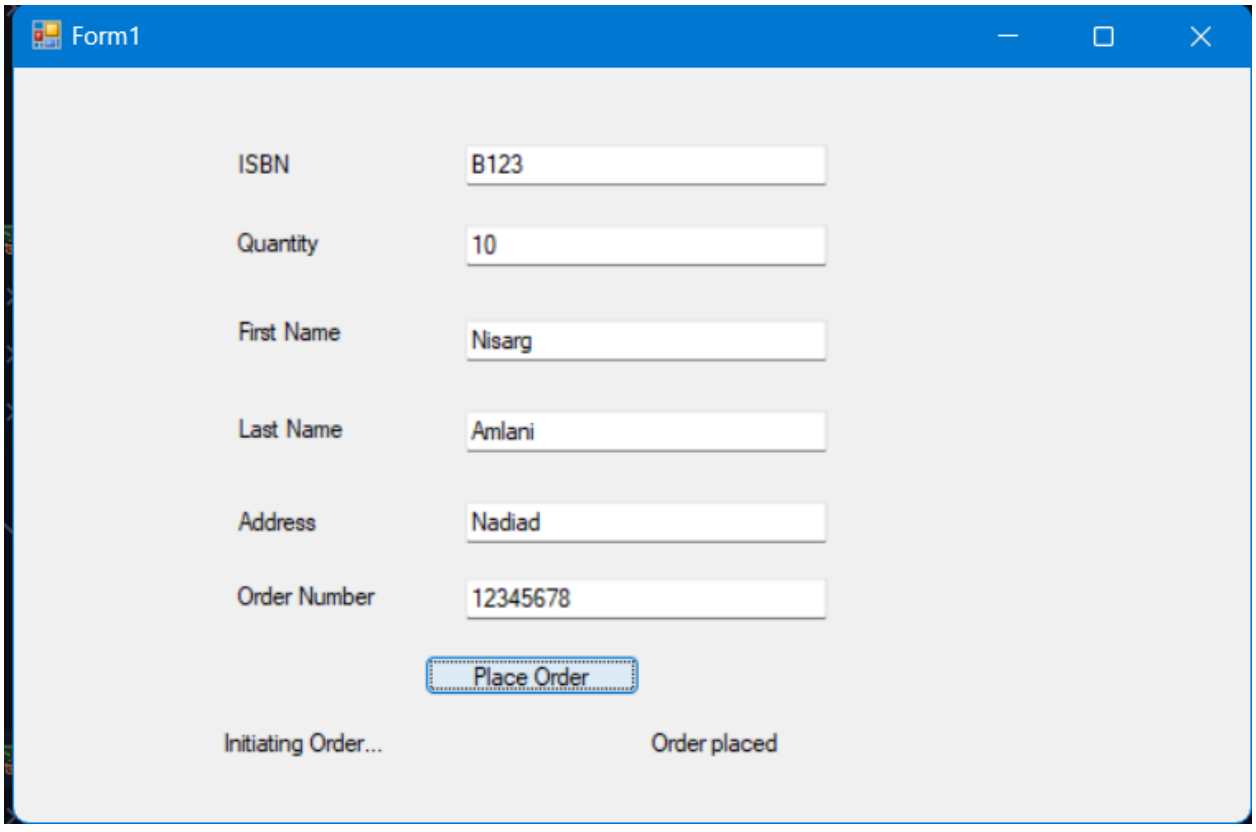
=> Form.cs (Client)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
using System.Windows.Forms;



namespace WindowsFormsAppClient
{
  public partial class Form1 : Form
  {
     public Form1()
     {
        InitializeComponent();
     }


     private void pl_order_Click_1(object sender, EventArgs e)
     {
        TCP.Service1Client client = new TCP.Service1Client();
        label7.Text = client.InitiateOrder();
        TCP.BookOrder order = new TCP.BookOrder();
        order.ISBN = textBox1.Text;
        order.Quantity = int.Parse(textBox2.Text);
        order.Firstname = textBox3.Text;
        order.Lastname = textBox4.Text;
        order.Address = textBox5.Text;


        TCP.BookOrder reply =          ((TCP.IService1)client).PlaceOrder(order);
        textBox6.Text = reply.OrderNumber;
        label8.Text = client.FinalizeOrder();
        client.Close();
     }
  }
}
```

=> Output (Client)


Form1
ISBN
B123
Quantity
10
First Name
Nisarg
Last Name
Amlani
Address
Nadiad
Order Number
Place Order
label7
label8


Form1
ISBN
B123
Quantity
10
First Name
Nisarg
Last Name
Amlani
Address
Nadiad
Order Number
12345678
Place Order
Initiating Order...
Order placed

Name : Nisarg .K.Amlani
Roll : Ce001
Id : 22ceueg082
Lab : 08

Web Api Code

=> Student.cs (Model Code)

```
using System.Diagnostics.CodeAnalysis;

namespace Web_Api_NetCore.Models
{
        public class Student
        {
        public int Id { get; set; }

        public string FirstName { get; set; }

        public string LastName { get; set; }

        public string Email { get; set; }

        }
}
```

=> StudentController.cs (Controller Code)

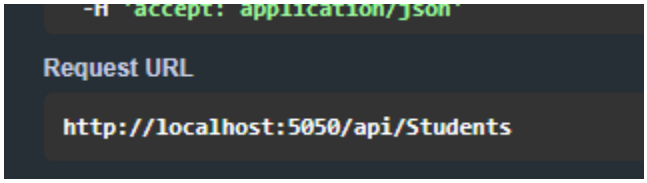
```
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
```

```
using Web_Api_NetCore.Models;

namespace Web_Api_NetCore.Controllers
{
	[Route("api/[controller]")]
	[ApiController]
	public class StudentsController : ControllerBase
	{
	private List<Student> _students = new List<Student>
	{
	new Student{Id =
1,FirstName="Nisarg",LastName="Amlani",Email="amlaninisarg15@gmail.com"},
	new Student{Id =
2,FirstName="Vaibhav",LastName="Dhanani",Email="vaibhavdhanani@gmail.co
m"},
	new Student{Id =
3,FirstName="Rich",LastName="Amrutiya",Email="richamrutiya@gmail.com"},
	new Student{Id =
4,FirstName="Mahek",LastName="Garala",Email="mahekgarala@gmail.com"}
	};

	[HttpGet]
	public ActionResult<IEnumerable<Student>> GetStudents()
	{
	return _students;
	}

	[HttpGet("{id}")]
	public ActionResult<Student> GetStudent(int id)
	{
	Student stud =  _students.FirstOrDefault(student => student.Id == id);
	if(stud == null)
		return NotFound();
	return stud;
	}

	[HttpPost]
```

```
        public ActionResult<Student> AddStudent(Student student)
        {
        _students.Add(student);
        return GetStudent(student.Id);

        }

        [HttpPut]
        public ActionResult<Student> UpdateStudent(int id , Student student)
        {
        Student toupdate = _students.FirstOrDefault(stud => stud.Id == id);
        if(toupdate == null) return NotFound();

        toupdate.FirstName = student.FirstName;
        toupdate.LastName = student.LastName;
        toupdate.Email = student.Email;
        _students.Add(toupdate);

        return GetStudent(toupdate.Id);
        }

        [HttpDelete("{id}")]
        public ActionResult DeleteStudent(int id)
        {
        Student toDelete = _students.FirstOrDefault(s  => s.Id == id);
        if(toDelete == null) return NotFound();

        _students.Remove(toDelete);
        return NoContent();
        }

        }
}
```

Output

=> Get All Students



200

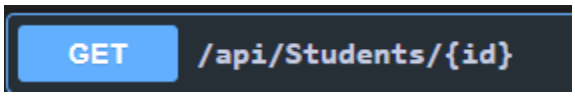
Response body


```
[
  {
    "id": 1,
    "firstName": "Nisarg",
    "lastName": "Amlani",
    "email": "amlaninisarg15@gmail.com"
  },
  {
    "id": 2,
    "firstName": "Vaibhav",
    "lastName": "Dhanani",
    "email": "vaibhavdhanani@gmail.com"
  },
  {
    "id": 3,
    "firstName": "Rich",
    "lastName": "Amrutiya",
    "email": "richamrutiya@gmail.com"
  },
  {
    "id": 4,
    "firstName": "Mahek",
    "lastName": "Garala",
    "email": "mahekgarala@gmail.com"
  }
]
```


Response headers

```
content-type: application/json; charset=utf-8
date: Wed,29 Jan 2025 22:16:55 GMT
server: Kestrel
transfer-encoding: chunked
```

=> Get Student by id



Request URL

```
http://localhost:5050/api/Students/1
```

Server response

Code	Details
200	Response body

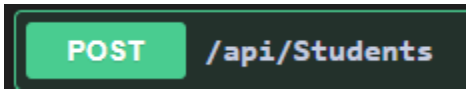
```
{
  "id": 1,
  "firstName": "Nisarg",
  "lastName": "Amlani",
  "email": "amlaninisarg15@gmail.com"
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Wed,29 Jan 2025 22:20:23 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

=> Add Student



Request body

```
{
  "id": 5,
  "firstName": "Nisarg",
  "lastName": "Amlani",
  "email": "22ceueg082"
}
```

Request URL

```
http://localhost:5050/api/Students
```

Server response

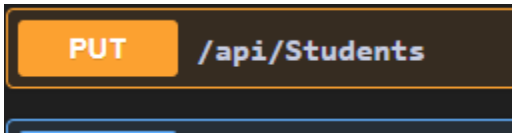
Code	Details
201 *Undocumented*	Response body

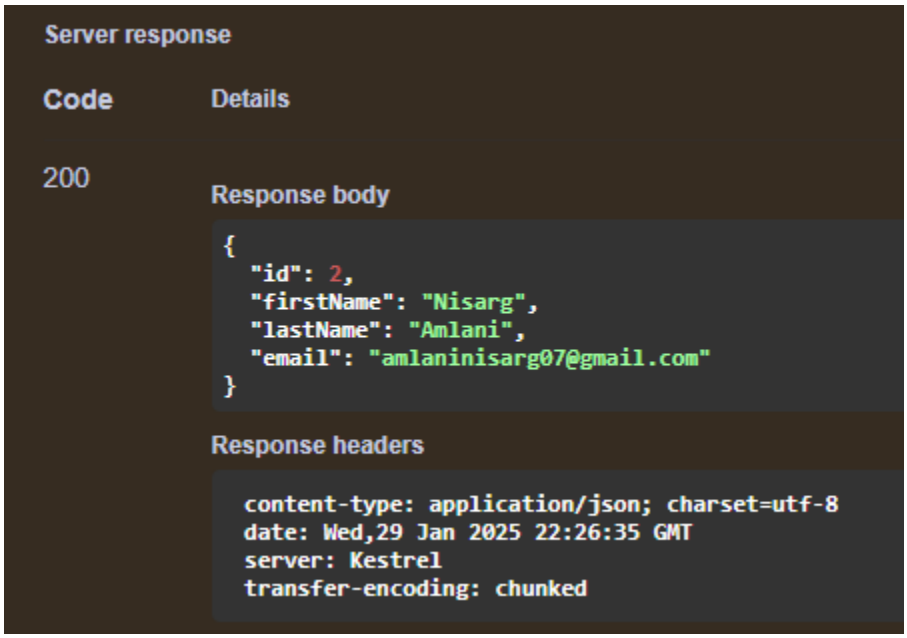
```
{
  "id": 5,
  "firstName": "Nisarg",
  "lastName": "Amlani",
  "email": "22ceueg082"
}
```

Response headers

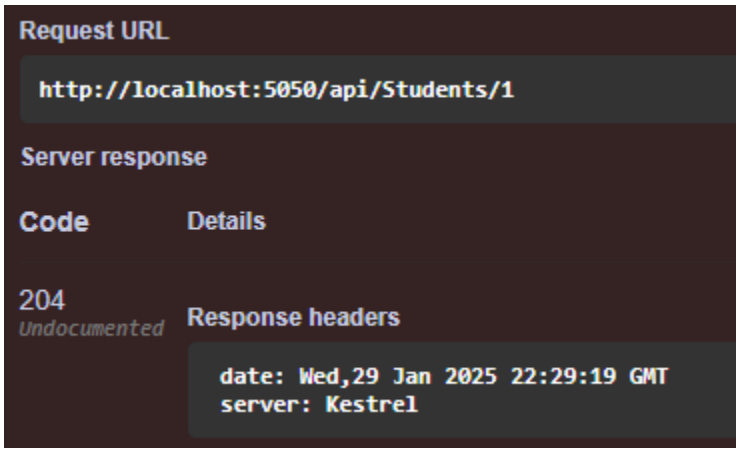
```
content-type: application/json; charset=utf-8
date: Wed,29 Jan 2025 22:24:49 GMT
location: http://localhost:5050/api/Students/5
server: Kestrel
transfer-encoding: chunked
```

=> Update Student





=> Delete the Student



Name : Nisarg .k. Amlani
Roll : Ce001
Id : 22ceueg082
Lab : 09

=> Student.cs Code

```
using System.ComponentModel.DataAnnotations;
namespace Lab_9.Models;

public class Student
{
  [Key]
  public int Id { get; set; }

  public string Name { get; set; }

  public string Email { get; set; }

  public int Sem {get; set;}
}
```

=> StudentContext.cs Code

```
using Microsoft.EntityFrameworkCore;

namespace Lab_9.Models;

public class StudentContext : DbContext
{
   public StudentContext(DbContextOptions<StudentContext>
options):base(options)
```

```
    { }

    public DbSet<Student> Students { get; set; }
}
```

=> Program.cs code

```
using Lab_9.Models;
using Microsoft.EntityFrameworkCore;
namespace Lab_9;
public class Program
{
  public static void Main(string[] args)
  {
     var builder = WebApplication.CreateBuilder(args);


     // Using In memory Database
     // builder.Services.AddDbContext<StudentContext>(opt =>
opt.UseInMemoryDatabase("StudentDB"));


     // Fetching Connection String
     var connectionString =
builder.Configuration.GetConnectionString("ConnectionString");
     // Using Database
     builder.Services.AddDbContext<StudentContext>(options =>
        options.UseMySql(
           connectionString,
           new MySqlServerVersion(new Version(10, 11, 10)) // Adjust to your MariaDB
version
        )
     );
```

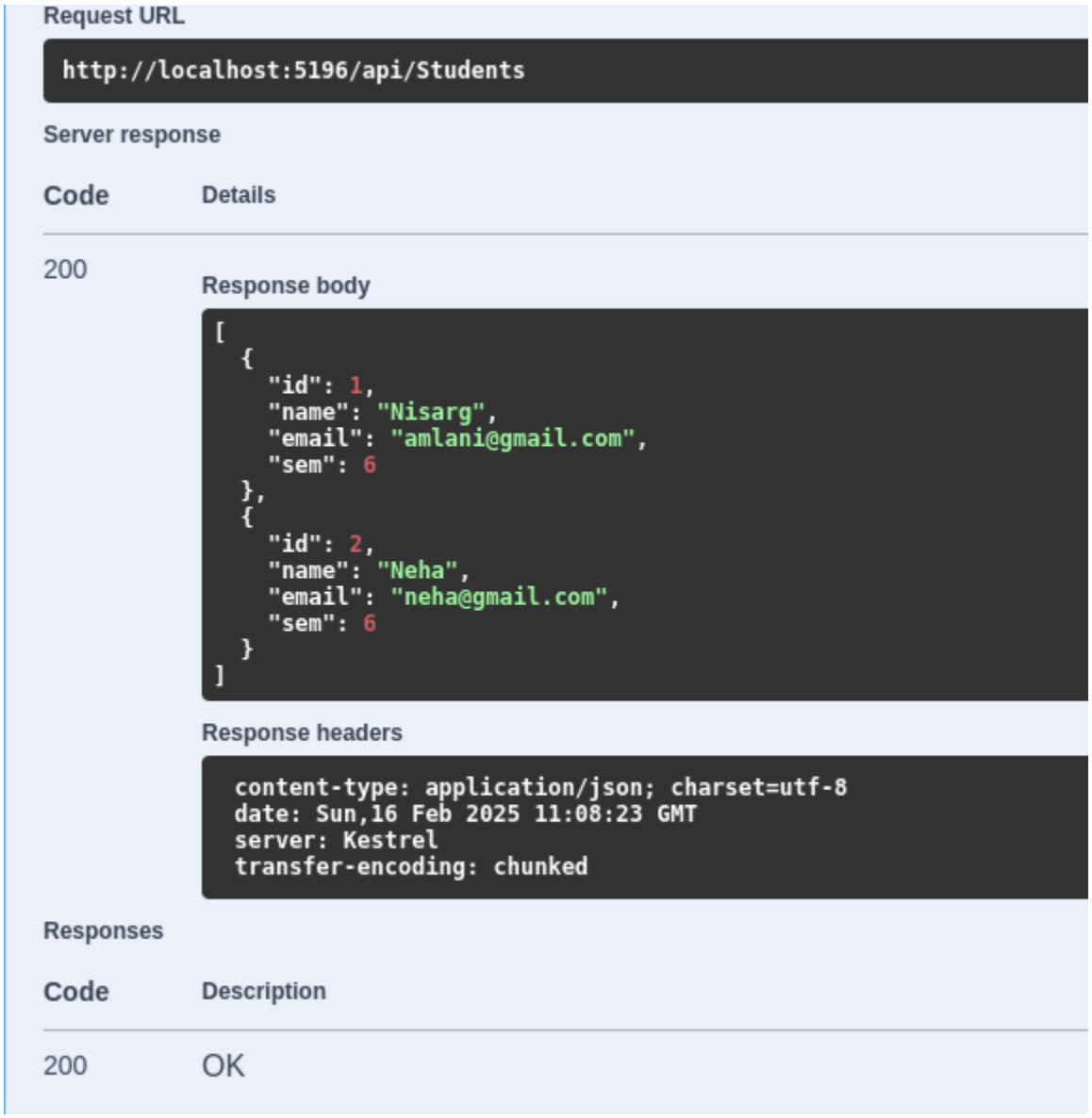
```
    // Add services to the container.


    builder.Services.AddControllers();
    // Learn more about configuring Swagger/OpenAPI at
https://aka.ms/aspnetcore/swashbuckle
    builder.Services.AddEndpointsApiExplorer();
    builder.Services.AddSwaggerGen();


    var app = builder.Build();
    // Configure the HTTP request pipeline.
    if (app.Environment.IsDevelopment())
    {
        app.UseSwagger();
        app.UseSwaggerUI();
    }
    app.UseHttpsRedirection();
    app.UseAuthorization();
    app.MapControllers();
    app.Run();
  }
}
```

=> appsetting.json code

```
{
 "Logging": {
   "LogLevel": {
     "Default": "Information",
     "Microsoft.AspNetCore": "Warning"
   }
 },
 "AllowedHosts": "*",
 "ConnectionStrings": {
   "Lab_9Context": "Data
Source=Lab_9Context-73761489-578a-4ab9-a0f9-493bdafef38f.db",
```

```
   "ConnectionString":
"Server=localhost;Database=WSD;User=nisarg;Password=nisarg2004;T
reatTinyAsBoolean=false;",
 }
}
```

Output Screenshots



Request URL

```
http://localhost:5196/api/Students
```

Server response

Code	Details
201 *Undocumented*	Response body

```
{
  "id": 2,
  "name": "Neha",
  "email": "neha@gmail.com",
  "sem": 6
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Sun,16 Feb 2025 11:08:08 GMT
location: http://localhost:5196/api/Students/2
server: Kestrel
transfer-encoding: chunked
```

Responses

Code	Description
200	OK

Request URL

```
http://localhost:5196/api/Students/2
```

Server response

Code	Details
200	Response body

```
{
  "id": 2,
  "name": "Neha",
  "email": "neha@gmail.com",
  "sem": 6
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Sun,16 Feb 2025 12:38:38 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

Code	Description
200	OK

Request URL

```
http://localhost:5196/api/Students/1
```

Server response

Code	Details

204
Undocumented
	Response headers
	`date: Sun,16 Feb 2025 12:40:09 GMT`
	`server: Kestrel`

Responses

Code	Description
200	OK

Request URL

```
http://localhost:5196/api/Students/1
```

Server response

Code	Details

204
Undocumented
	Response headers
	`date: Sun,16 Feb 2025 12:40:44 GMT`
	`server: Kestrel`

Responses

Code	Description
200	OK