Create a simplified messaging service to buffer messages in an XML file and a Web client to allow users to send and receive messages. You will develop your service and application on IIS Express localhost.

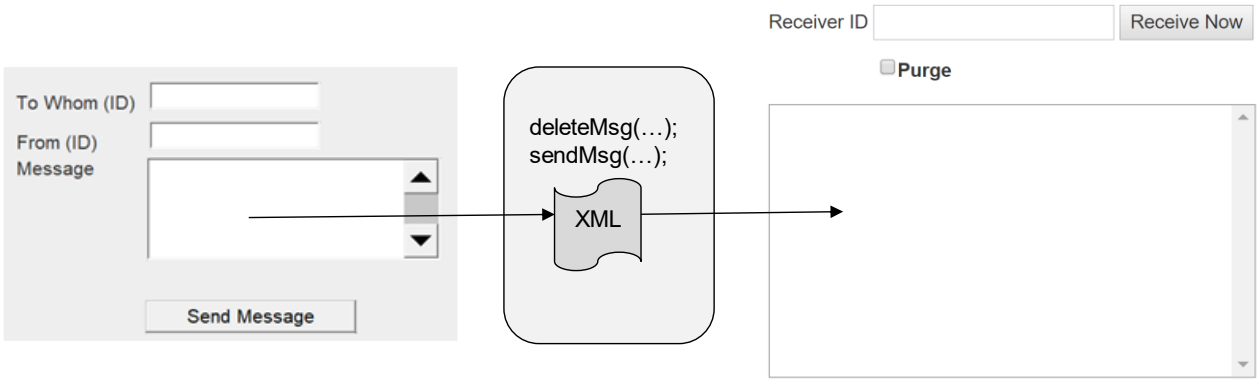
1 Messaging service: Develop a service (WSDL service, RESTful service, or Workflow service) that can buffer messages before the receivers fetch the messages. The messages must be saved in an XML file (or JSON file) that the service can access. The service must offer the following operations. You may add more parameters for additional functions. [25]

- Operation 1: sendMsg: It allows the client to send a string message to the messaging service, and the message will be stored in the database (XML file) with senderID, receiverID, and a time stamp.
  Inputs: string senderID, string receiverID, string msg.
  Output: void
  Note: a timestamp will be added by the program.

- Operation 2: receiveMsg: It allows the client to receive all the messages send to the receiverID.
  Inputs: string receiverID, boolean purge
  Output: string[] an array of messages, with each element containing the related information: senderID, sending time, and message.
  Note: The receiver should receive the new messages that have not been received in the previous receive call. If purge == true, the service will delete all messages of the receiver.

To read and write an XML file on disk, you may want to read text Chapter 4 on XML processing and Section 5.4.2 on reading and writing XML files. You may also read Text Chapter 10 and using LINQ to XML methods.

2 MsgApp: Develop a Web application (client) in ASP .Net or MVC that contains at least two Web pages: Sender page and Receiver page. The sender sends messages to the Messaging service and the receiver receives messages from the service. A sample GUI is shown below. The GUI must link to all components that need to be implemented. You can add additional items in your GUI. [25]




To test your clients (sender and receiver) on localhost/IIS Express: Right click the service project and choose View in Browser to run the service. Then, start the client, which will open another bowser window to run the client. Copy and paste the URL into a new browser window. Now, you have three sessions of your application: one service session and two client sessions. You can test your application by sending a message in one browser and receiving the message in another browser. Submit a screenshot of your testing.