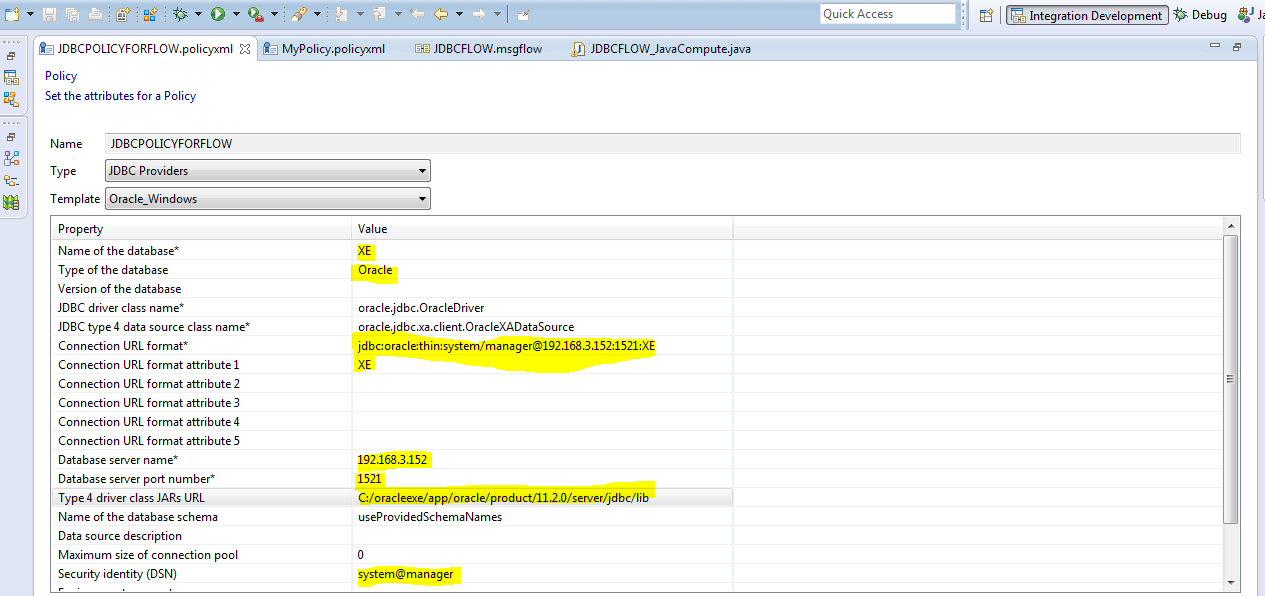
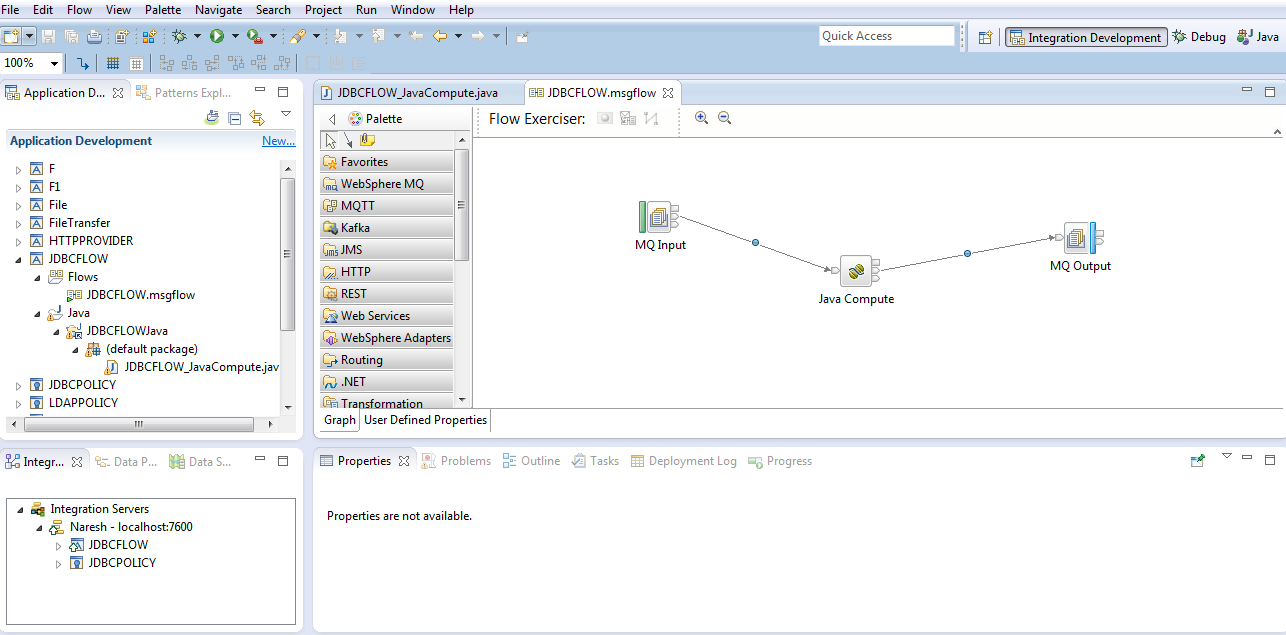
**JDBC PROVIDER:**

****

****

**import** java.sql.Connection;

**import** java.sql.ResultSet;

**import** java.sql.Statement;

**import** com.ibm.broker.javacompute.MbJavaComputeNode;

**import** com.ibm.broker.plugin.MbElement;

**import** com.ibm.broker.plugin.MbException;

**import** com.ibm.broker.plugin.MbMessage;

**import** com.ibm.broker.plugin.MbMessageAssembly;

**import** com.ibm.broker.plugin.MbOutputTerminal;

**import** com.ibm.broker.plugin.MbUserException;

**import** com.ibm.broker.plugin.MbXMLNSC;

**public** **class** JDBCFLOW\_JavaCompute **extends** MbJavaComputeNode {

**public** **void** evaluate(MbMessageAssembly inAssembly) **throws** MbException {

MbOutputTerminal out = getOutputTerminal("out");

MbOutputTerminal alt = getOutputTerminal("alternate");

MbMessage inMessage = inAssembly.getMessage();

// create new empty message

MbMessage outMessage = **new** MbMessage();

MbMessageAssembly outAssembly = **new** MbMessageAssembly(inAssembly,

outMessage);

**try** {

// optionally copy message headers

copyMessageHeaders(inMessage, outMessage);

// ----------------------------------------------------------

// Add user code below

Connection con=**null**;

Statement st=**null**;

con=getJDBCType4Connection("JDBCPOLICYFORFLOW", JDBC\_TransactionType.*MB\_TRANSACTION\_AUTO*);

policy name

st=con.createStatement();

ResultSet rs=st.executeQuery("select \* from eidiko\_employee");

MbElement root=outMessage.getRootElement();

MbElement parser=root.createElementAsLastChild(MbXMLNSC.*PARSER\_NAME*);

MbElement firstChild = parser.createElementAsLastChild(MbElement.*TYPE\_NAME*,"firstChild",**null**);

**while**(rs.next()){

**int** eid = rs.getInt("EMPID");

String ename = rs.getString("NAME");

String eaddr = rs.getString("ADDR");

firstChild.createElementAsLastChild(MbElement.*TYPE\_NAME*,"EID",eid);

firstChild.createElementAsLastChild(MbElement.*TYPE\_NAME*,"ename",ename);

firstChild.createElementAsLastChild(MbElement.*TYPE\_NAME*,"eaddr",eaddr);

}

// End of user code

// ----------------------------------------------------------

} **catch** (MbException e) {

// Re-throw to allow Broker handling of MbException

**throw** e;

} **catch** (RuntimeException e) {

// Re-throw to allow Broker handling of RuntimeException

**throw** e;

} **catch** (Exception e) {

// Consider replacing Exception with type(s) thrown by user code

// Example handling ensures all exceptions are re-thrown to be handled in the flow

**throw** **new** MbUserException(**this**, "evaluate()", "", "", e.toString(),

**null**);

}

// The following should only be changed

// if not propagating message to the 'out' terminal

out.propagate(outAssembly);

}

**public** **void** copyMessageHeaders(MbMessage inMessage, MbMessage outMessage)

**throws** MbException {

MbElement outRoot = outMessage.getRootElement();

// iterate though the headers starting with the first child of the root

// element

MbElement header = inMessage.getRootElement().getFirstChild();

**while** (header != **null** && header.getNextSibling() != **null**) // stop before

// the last

// child

// (body)

{

// copy the header and add it to the out message

outRoot.addAsLastChild(header.copy());

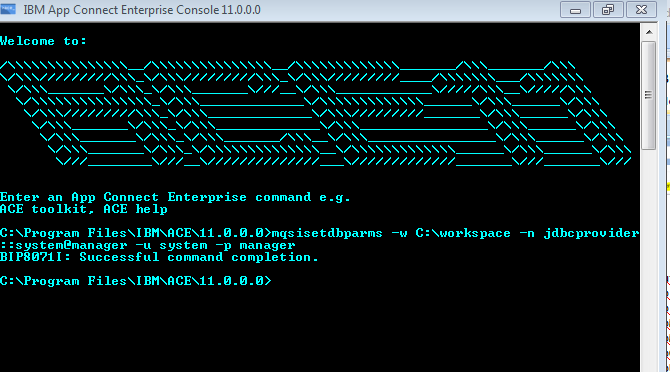
// move along to next header

header = header.getNextSibling();

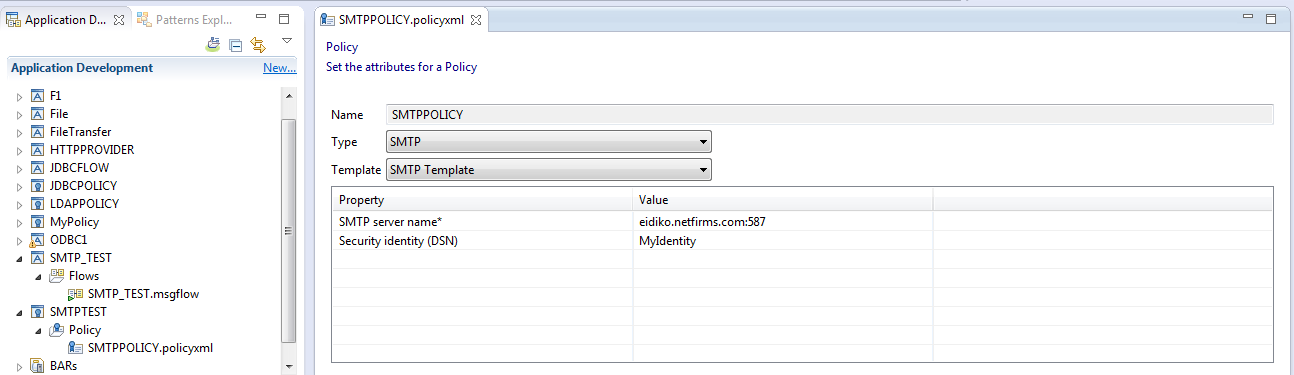
}

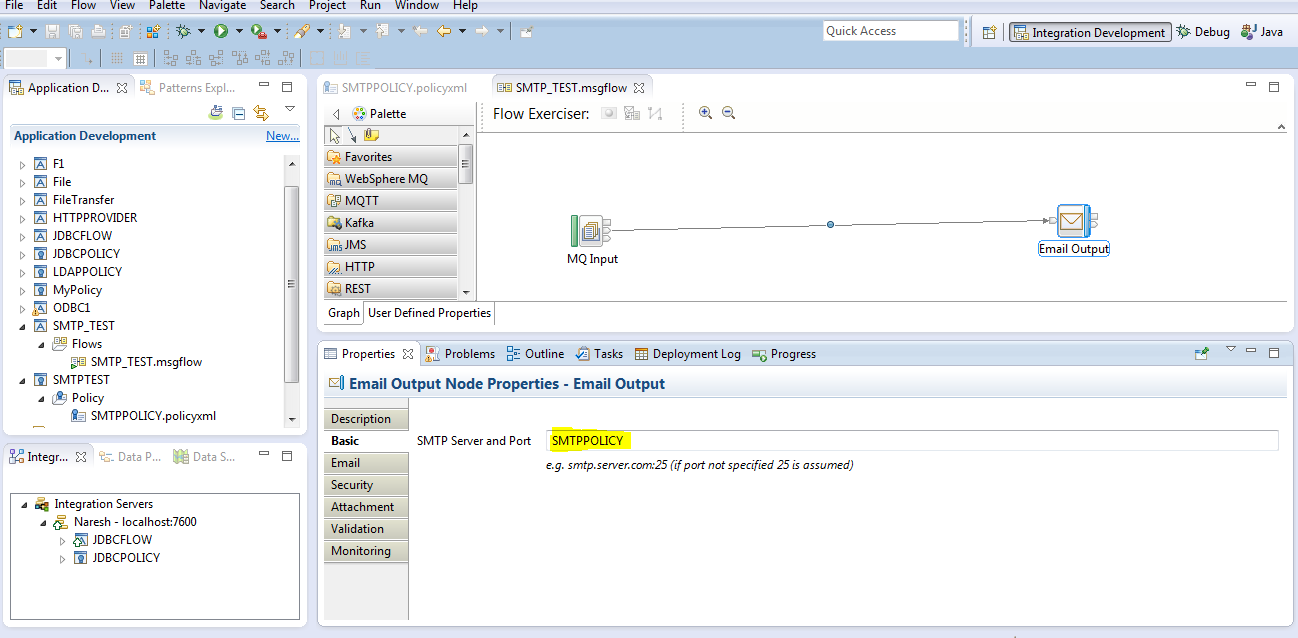
}

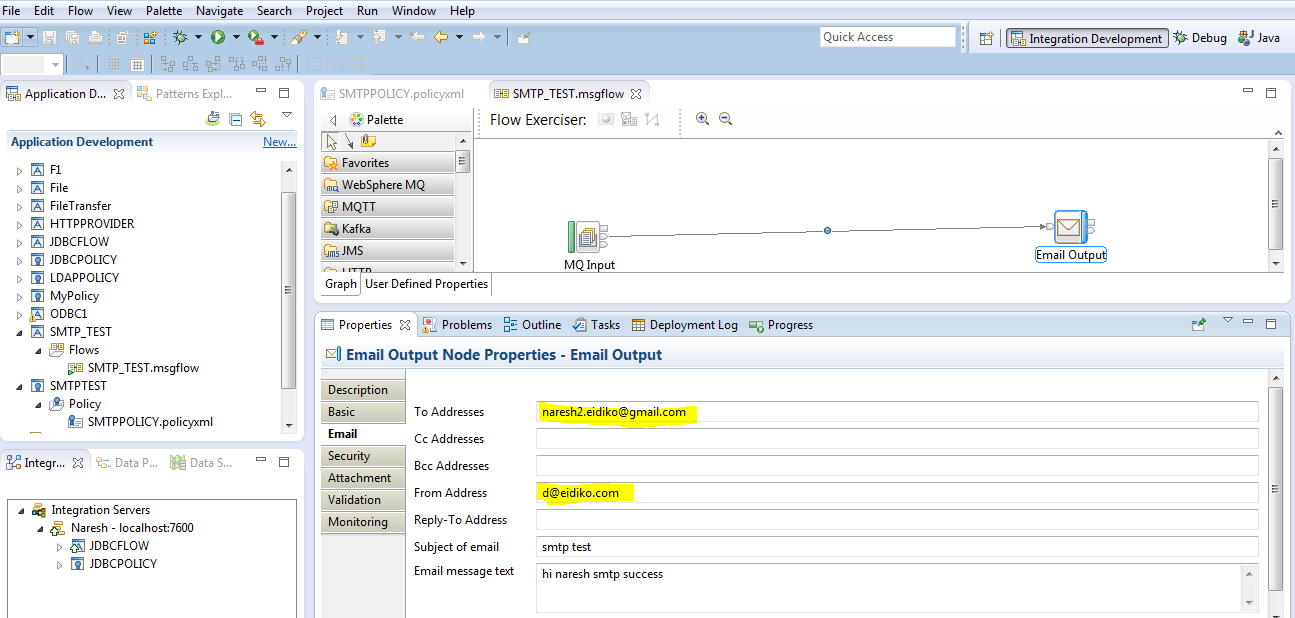
}



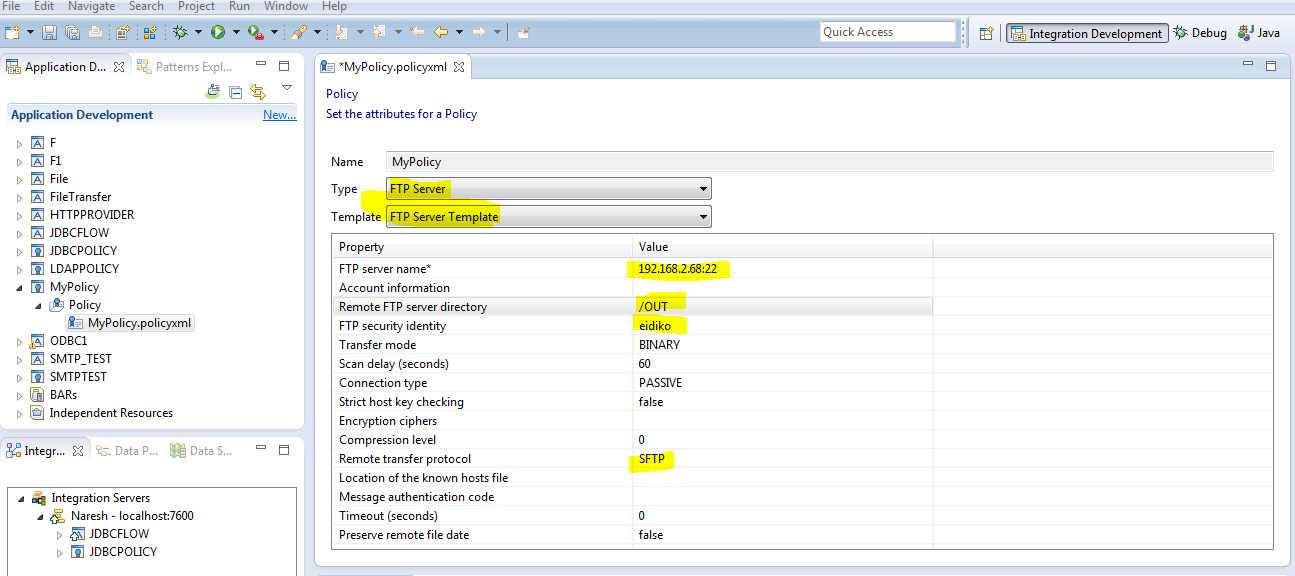
**SMTP:**

mqsisetdbparms -w C:\workspace -n smtp::MyIdentity -u naresh.nakka@gmail.com -p Naresh@123

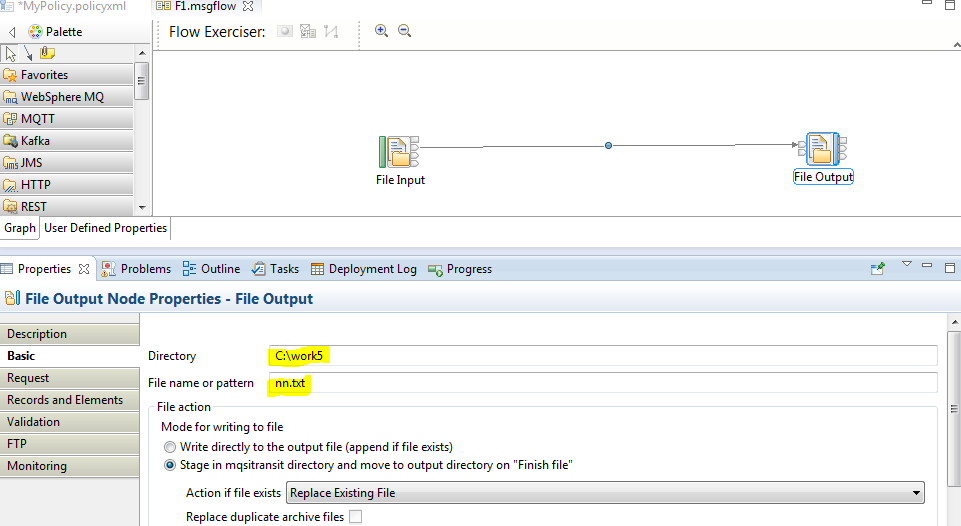


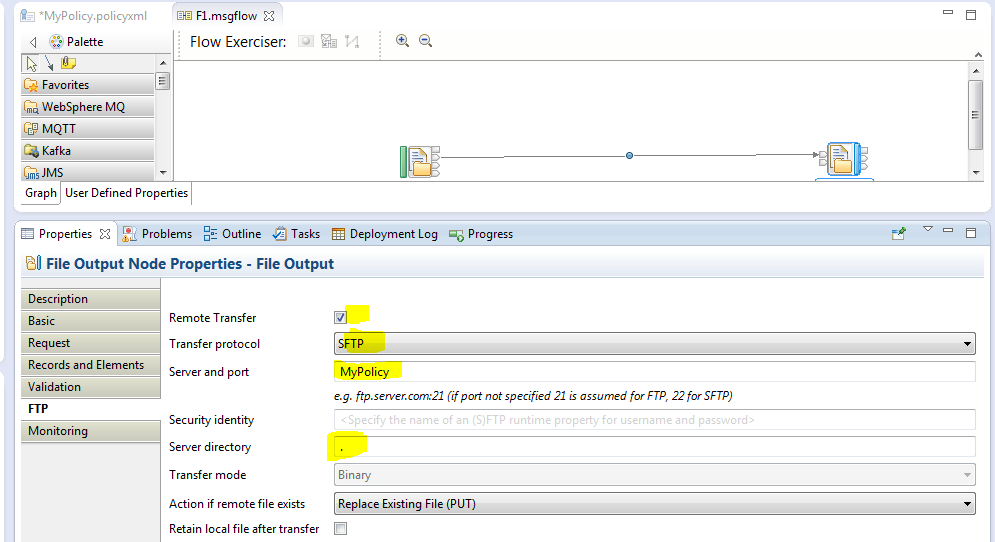


**SFTP:**

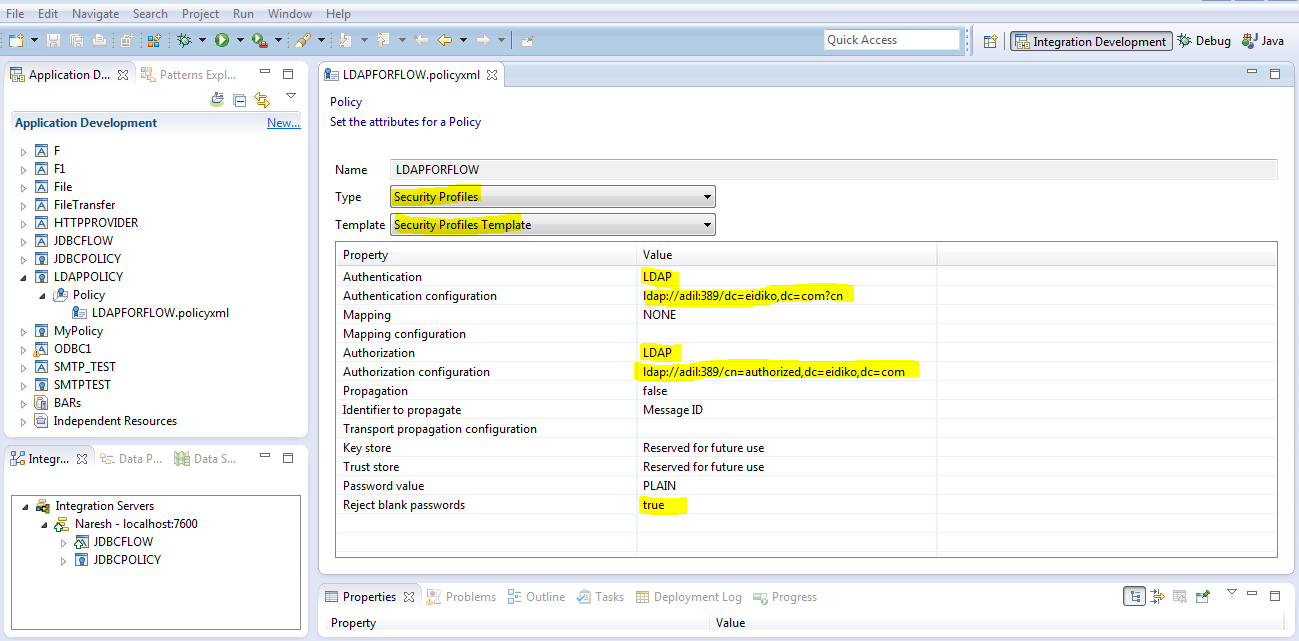


mqsisetdbparms -w C:\workspace -n sftp::eidiko -u bandaru -p sarasu10

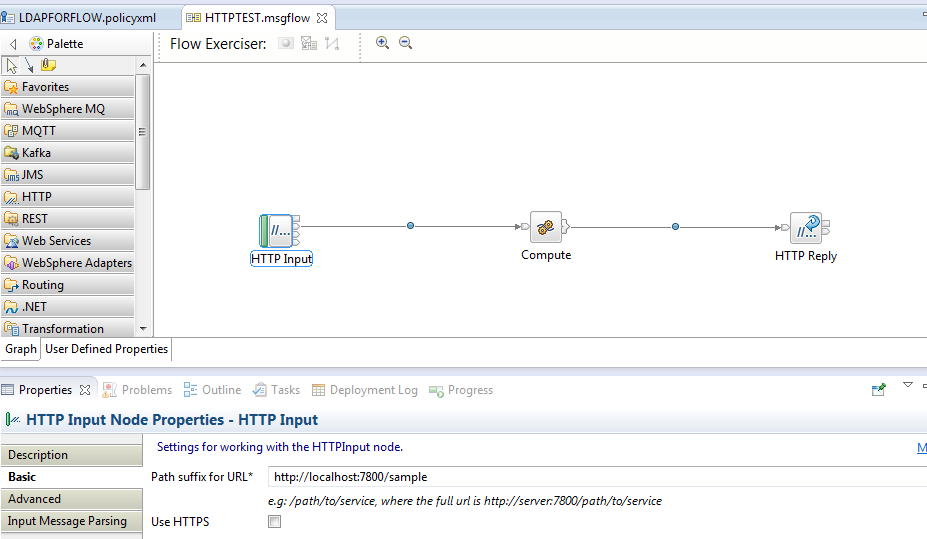


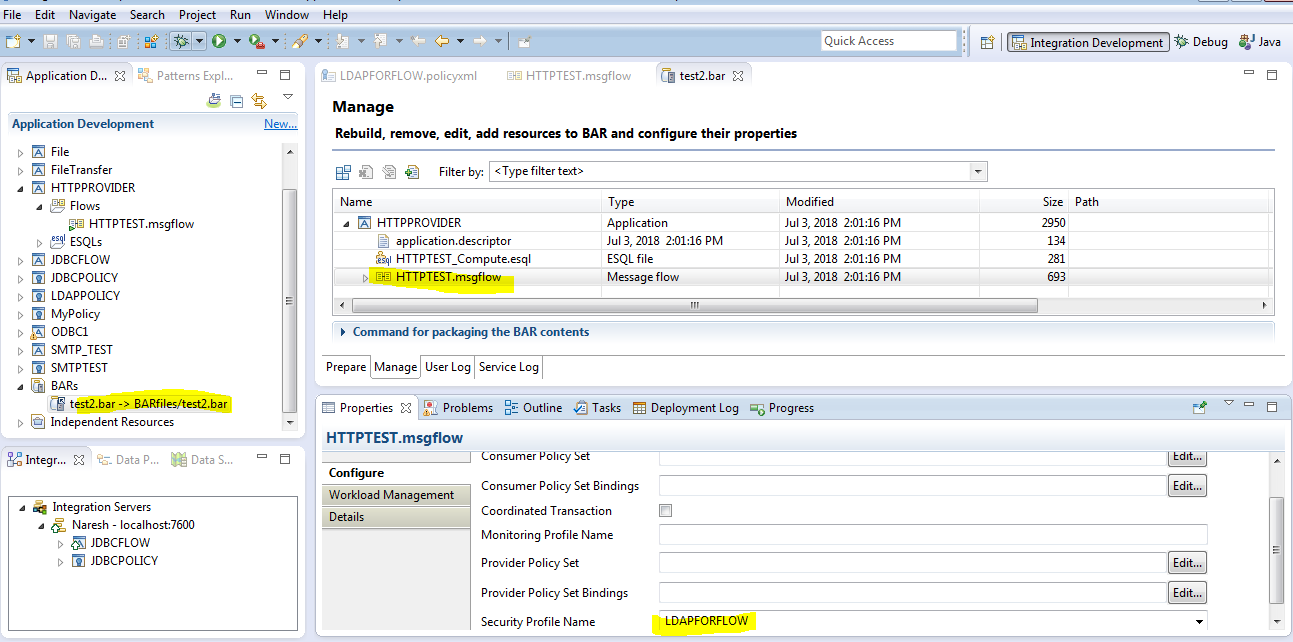


**LDAP:**

****

**mqsisetdbparms -w C:\workspace -n ladp::adil -u "cn=adil,dc=eidiko,dc=com" -p sarasu10**

****

****