1. About OYAK Marketing Service and Tourism Co.

OYAK has been established by members of Turkish Army Forces to provides additional security for any social, economical and physical risks in 1th March 1961. The total members were around sixty five thousand active workers.

In the same year, the company has started to support money for its members and developed several social service applications.

In 1998, the company adapted itself for needs of market. Therefore, OYAK Marketing has been set up as a subcompany of OYAK.

In next step, OYAK Tourism united with OYAK Marketing under the name of OYAK Marketing. Following years, other companies have exhanged working areas with OYAK Marketing. The company kept growing and had regeneration due to need of adaptation.

In 2014, Oyak Technology where I did my intern has joined to OYAK Marketing and till now, developed software and store database for other companies.

The firm has projects on Java, HTML, SQL, PL/SQL, Oracle and .NET. The company generally collaborates with bank companies and serves software for other firms that belong to OYAK. Depend on the demands, Oyak Technology developes individual softwares for companies or private requests.

2. Technologies Used In The Training

First of all, I have used three technology to develop my first Web application. The first one was Apache Axis2. It is essential because it is a core engine for Web Services. It is a complete re-design and re-write of the widely used Apache Axis SOAP stack. Implementations of Axis2 are available in Java and C. Axis2 provides the capability to add Web services interfaces to Web Applications. It can also function as a standalone server application. Furthermore, I installed Apache Tomcat 7.0. Apache Tomcat, often referred to as Tomcat, is an open-sourceweb server developed by the Apache Software Foundation. Tomcat implements several Java EE specifications including Java Servlet, JavaServer Pages, etc. The most important point is that it provides a "pure Java" http web server environment in which Java code can run.

Thanks to Tomcat, I was able to construct my server successfully and test it in the web environment. Without it, I couldn't see what my web application's condition is.

Another program is SOAPUI. SOAPUI is an open-source web service application for service-oriented architectures(SOA) and representational state transfers(REST). I tested my methods that I have written. In order to observe the results, my methods were returning information. I created a new Project and gave a name and a WSDL then press OK, it generates and I was controlling my web services without any problem.

Later days, I read tutorials for SQL and database management. Big data are accessed and managed easily with database systems. I used SQL Server 2008 R2. I created a new server and a new database then by using several syntax, I was capable of inserting, upgrading, deleting and reading the data.

My IDE was Eclipse Juno. I used to use Eclipse Luna during my semester and when I started my intern, I supposed Eclipse versions doesn't make a huge difference. I noticed, I was wrong and I installed Eclipse Juno to make my Java applications.

3. Work Done During The Intern

a. Week 1

First week, I mostly spent my time on learning general information about Web and Web Services. My instructor wants me to deal with some basic definitions and get the general logic of web services. In order to achieve this, I searched a lot of topics on the internet such as:

-What is Web Service and its logic?

-What is WSDL, XML files?

-How different programming languages interact each other?

-What is Servlet Container?

-Main parts of modern internet & Web System (URL, HTTP, IP)?

Once I got those useful stuff, I decided to initiate a simple project which related to Web Service and Server.

The sample was on this link:

http://www.eclipse.org/webtools/community/education/web/t320/Implementing_a_Simple_Web_Service.pdf

b. Week 2

This week, I started my project. I used Eclipse Juno for both Java and Java EE development. Before struggling with tough ones, my instructor wanted me to deal with a simple one. At first, I tried to install JBoss v4.2.0 but I couldn't starting the server. Constructing server has taken so much time. On Thursday, I described the situation that I was into and later, me and my instructor told me to find another server application. Because, it was essential to make any web service. Finally, I found Apache Tomcat Server v7.0 and succesfully started on Friday.

Besides, I have learnt server and client logic. How internet sites work properly and what kind of things are working behind the page. I can now distinguish client side and server side operations. There are programming languages using for client side programming and other side, for server side programming. In order to send a request to web page, initially I was adding piece of code to my project and whenever I want to make a change, I have to stop the server and restart again and again. Naturally, it takes so much time and drained my patience. Therefore, I decided to find whether any available program exist or not. Fortunately, there was. My instructor said to me that I could use SOAPUI. It is an open-source program and was really easy to use.

There were 2 main frame I can set my input and get the output. A screenshot is taken for demonstration below.
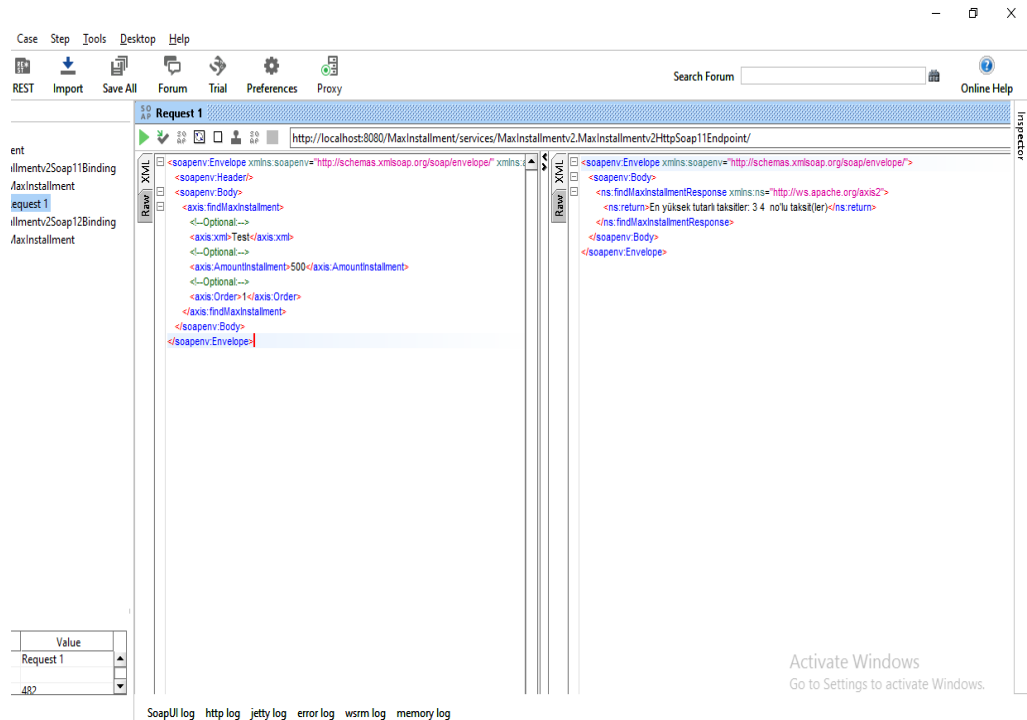
Next step was inserting a WSDL file into Eclipse which has been generated by the web service. I have learnt once you get a WSDL file then you were able to transform it to java code which consist of classes and methods. There was an useful tool for that operation. As creating a new project, anyone may select and simply following the steps he/she can import WSDL files.

End of this week, on Friday, I was asked to make a web service that gets input and calculates the highest installment among all of them and then returns output. I used 2 classes for this procedure which are Kredi.java and MaxInstallmentv2.java.

The libraries is used in this project below:

MaxInstallment.java Class Imported Library

java.io.BufferedReader, java.io.File, java.io.FileReader, java.io.IOException, java.io.Reader, java.io.StringReader, java.sql.Connection, java.sql.DriverManager, java.sql.PreparedStatement, java.sql.SQLException, java.util.ArrayList, java.util.List, javax.xml.bind.JAXBContext, javax.xml.bind.JAXBElement, javax.xml.bind.JAXBException, javax.xml.transform.stream.StreamSource

Kredi.java Class Imported Library

java.util.ArrayList, java.util.List

At very first, I had figured it out with 3 inputs. There were 3 components which order, payment date, amount of the installment. I should have returned the output in the following format:

'En yüksek tutarlı taksitler A ve B no'lu taksit(ler).'

After succesfully achiving this, I focused on upgrading my project to a better level but after weekend.

c. Week 3

As I considered, I could make an improvement. I thought I could make an array list for large amount of inputs. Although this part was the most diffucult part for this project, I did it. Instead every time giving inputs, my program was able to read data from a TXT file which needed a certain format. This format should has been a list of elements with 3 paramaters. In addition, this improvement could give me more time for test and fix.

Following day, I had another task. The task was converting the input file format from TXT to XML. As far as I have seen on the internet, there were several libraries that already in use which:

javax.xml.parsers.DocumentBuilder,
javax.xml.parsers.DocumentBuilderFactory

These classes are for the exceptions that can be thrown when the XML document is parsed:

- org.xml.sax.ErrorHandler,
- org.xml.sax.SAXException,
- org.xml.sax.SAXParseException,
- org.xml.sax.helpers.*

However, I prefered my own way. My way was split the xml file and read every single installment then put it inside the array list. This milestone gave me the opportunity to how to use Java array list library and to do simple operations. For example, I know how to insert an element into arraylist or delete, upgrade it inside the array list. Moreover, I know how to access a

particular array list member. It takes 4 days to complete all these developments. The codes is going to show in the following pages.

MaxInstallment project's WSDL was this:

This week, I also learnt the principals of SQL Database logic and set up SQL Server on my computer. I have used http://www.w3schools.com/sql/default.asp for SQL lessons.

d. Week 4

After getting more stuff about SQL, I installed Microsoft SQL Server 2008. Oyak has been mostly using SQL and Oracle databases for its business, therefore problems about database solved rapidly and most of my work turned to database operations and database interactions between SQL database and web services. My new task was reading XML files and inserting data to my SQL server database. My first job was connecting SQL database and Eclipse each other. I need to make a test to see the connection. I created a class DatabaseTest.java I did use available libraries in my class which are:

java.sql.Connection, java.sql.DriverManager, java.sql.ResultSet,
java.sql.SQLException, java.sql.Statement

The libraries allowed me to use whole SQL commands in my java class.
I listed my sample database elements to be sure SQL – Java connection has
established. Here is the result of this operation:
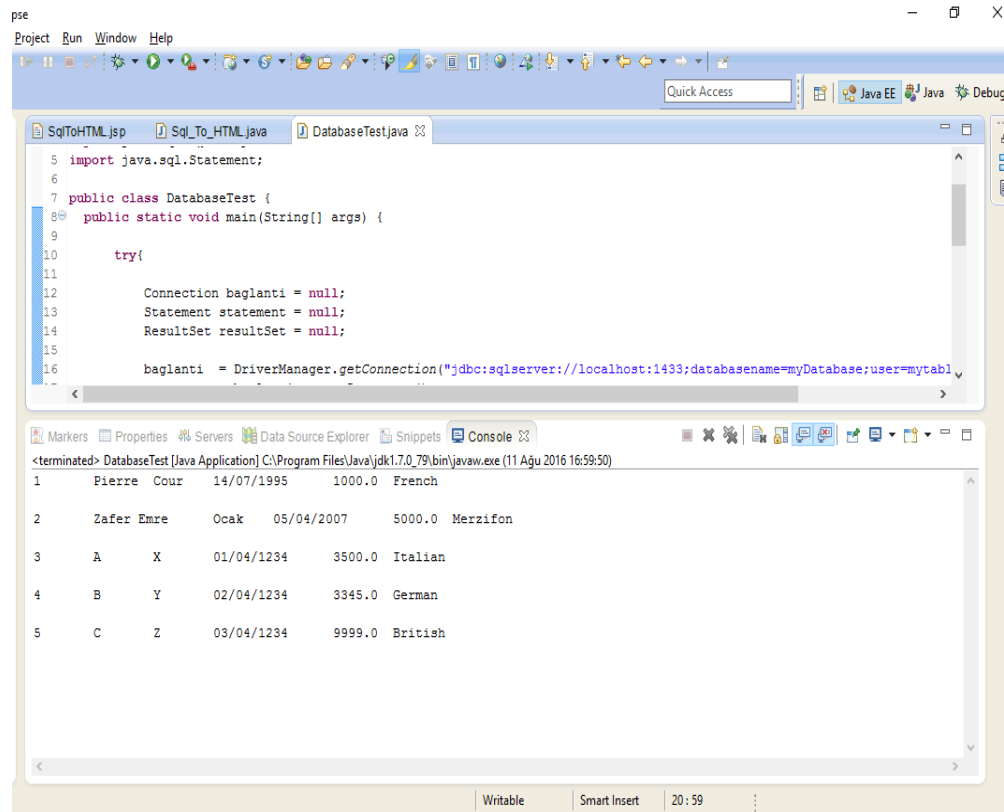
Also, codes is here where placed on MaxInstallmentv2.java:

List<Integer> maxCredits = null;

int i = 0,j = 0;

String output = "";

String SQL_INSERT = "INSERT INTO Table_Kredi VALUES(?,?,?)";

String SQL_UPDATE = "UPDATE Table_Kredi SET Taksit_Tutari=? WHERE Sira=?";

try{

String info = "";

//Kredi arrayList created

```
ArrayList<Kredi> Credits = new ArrayList<Kredi>();


File xmlFile = new File("C:/yazilim/" + xml + ".xml");

Reader fileReader = new FileReader(xmlFile);

BufferedReader bufReader = new BufferedReader(fileReader);

StringBuilder sb = new StringBuilder();

String line = bufReader.readLine();

while(line != null){

        sb.append(line);

        line = bufReader.readLine();

}

String xmlString = sb.toString();

bufReader.close();

JAXBContext jc = JAXBContext.newInstance(Kredi.class);

javax.xml.bind.Unmarshaller unmarshaller = jc.createUnmarshaller();

//SQL Binding

Connection baglanti = null;

PreparedStatement statement = null;


Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");


baglanti =
DriverManager.getConnection("jdbc:sqlserver://localhost:1433;databas
ename=myDatabase;user=mytable;password=emre3673");

statement = baglanti.prepareStatement(SQL_INSERT);
```

```java
for(String s : xmlString.split("(?=<sira>)|(?<=</taksit_tutari>)")){

    i++;

    if((i%2)==0 && (i!=0)){

        info = "<Kredi>"+s+"</Kredi>";

        String info11 = info.replaceAll("[ ]", "");



StreamSource streamSource = new StreamSource(newStringReader(info11));

JAXBElement<Kredi> je = unmarshaller.unmarshal(streamSource,Kredi.class);


        Kredi kredi = new Kredi();

        kredi = (Kredi)je.getValue();

        //INSERT TO TABLE Table_Kredi

        statement.setInt(1, kredi.getSira());

        statement.setString(2, kredi.getOdeme_tarihi());

        statement.setInt(3, kredi.getTaksit_tutari());

        statement.executeUpdate();

        Credits.add(j, kredi);

        j++;

        }

                    }

    PreparedStatement updateStatement = null;

    updateStatement = baglanti.prepareStatement(SQL_UPDATE);

    updateStatement.setInt(1, AmountInstallment);

    updateStatement.setInt(2, Order);

    updateStatement.executeUpdate();
```

I have gone well after these 2 weeks passed. The project was pretty simple and it was only able to read data from TXT file and return the maximum installment. This task is actually very significant for me to understand how things work on background. Entire effort just makes this process simplier and faster. Now, I can see the importance of computers and their results.

e. Week 5

Fifth week, I spent my first day to JSP(Java Server Pages). I searched Google and found useful information. I looked over these links especially:

- http://www.w3schools.com/html/default.asp
- http://www.tutorialspoint.com/jsp/
- http://www.apekshit.com/JSP-Tutorial-for-beginners-with-Examples/c/21

JSP is similar to HTML but it allows you to write piece of java codes inside it where you must use "<% %>" brackets. Also, the developer could make scripts to make pages more dynamic and visual.

For example, my database's elements were consisting of 6 parameters and additionally, there were edit and done buttons to make changes and upgrading the elements.

```
<%

        for(i=0; i<IDList.size(); i++){

        %>

        <tr>

        <td>   <input type=text disabled="disabled" id="id<%=i%>"
                value=<%=IDList.get(i)%> >  </td>

        <td>   <input type=text disabled="disabled" id="name<%=i%>"
        value=<%=NameList.get(i)%> > </td>

        <td>   <input type=text disabled="disabled"
        id="surname<%=i%>"
        value=<%=SurnameList.get(i)%> > </td>
```

```
                    <td>  <input type=text disabled="disabled"
id="birthdate<%=i%>"          value=<%=BirthdateList.get(i)%> > </td>

                    <td>  <input type=text disabled="disabled"
id="salary<%=i%>"           value=<%=SalaryList.get(i)%> > </td>

                    <td>  <input type=text disabled="disabled"
id="nationality<%=i%>"      value=<%=NationalityList.get(i)%> > </td>

                    <td>  <button type="button" id="edit<%=i%>"
        onclick="Edit(<%=i%>)">Edit</button></td>

                    <td>  <button type="button" id="confirm<%=i%>"
onclick="Confirm(<%=i%>)">Done</button></td>

                    </tr>

                    <%

                    }

        %>
```

This code has been written by me. It gives the user the opportunity of edit and set elements. At the end, database upgrades itself depending on these new data.

Furthermore, I worked on JavaScript in my project. To make visible the effects of buttons like edit and done, I was need to write a script. At first, I visited http://www.w3schools.com/js/default.asp.

A script pattern which was used in my project:

```
<SCRIPT>

function Edit(i){

        if(i == undefined){

                i = 0;

        }

        document.getElementById("id"+i).disabled = false;
```

document.getElementById("name"+i).disabled = false;

document.getElementById("surname"+i).disabled = false;

document.getElementById("birthdate"+i).disabled = false;

document.getElementById("salary"+i).disabled = false;

document.getElementById("nationality"+i).disabled = false;

}

</SCRIPT>

This script prevents user to edit sections. To make everything clear, I am going to put an image to show their functions.

f.   Week 6

Final week… I was at the point where to submit new elements to SQL database. To begin with, I have been looking for how to reverse the procedure to receive data from web page and play with them for changes. I had not known "Servlets" before realize how easy is. Servlet is used for sending request to web pages and waits for responds to do anything. I visited http://www.tutorialspoint.com/servlets/ to get information about servlets.

Once I achieved how to use servlets, all I should do was connecting my web page and my servlet to each other. Then, I succesfully completed connecting operation. I created 2 methods for submit/upgrade operation as they have been shown below:

//METHOD TO SUBMIT

public void clickUpdate(HttpServletRequest request, HttpServletRequest response) throws ServletException, IOException, SQLException{

Sql_To_HTML sql_to_html = new Sql_To_HTML();

String button = request.getParameter("button");

if("button1".equals(button)){

sql_to_html.update();

        }

    }

//METHOD FOR SQL COMMAND

public void update() throws SQLException{

resultSet = statement.executeQuery("UPDATE myDatabase Set ID='ID' WHERE ID=ID");
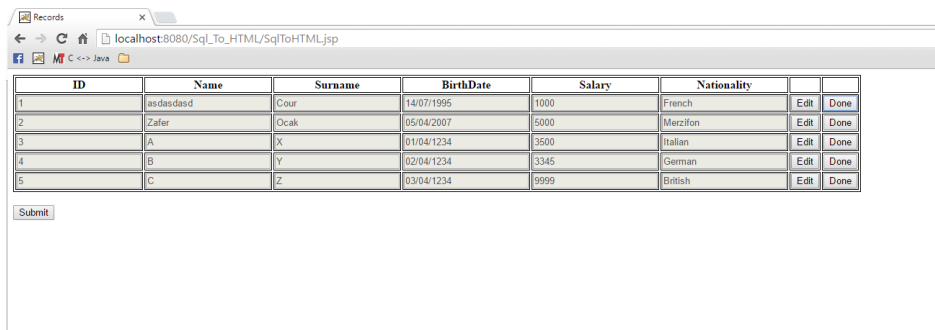
    }

Inside "clickUpdate", after pressing submit button, it compares done buttons and decided which one is going to be changed. If it finds a matched pair then method "update" appears and executes the SQL statement to update the specific data properly.
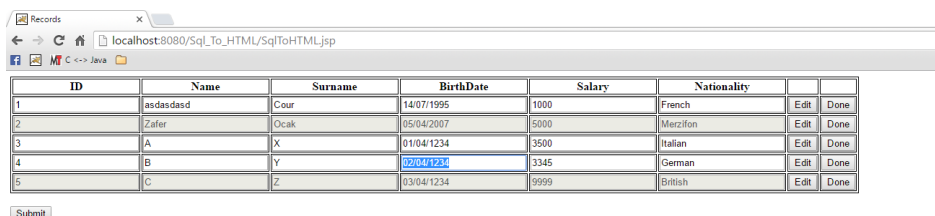
There are 3 pictures to make things clear.

First picture shows the general frame of my web page:
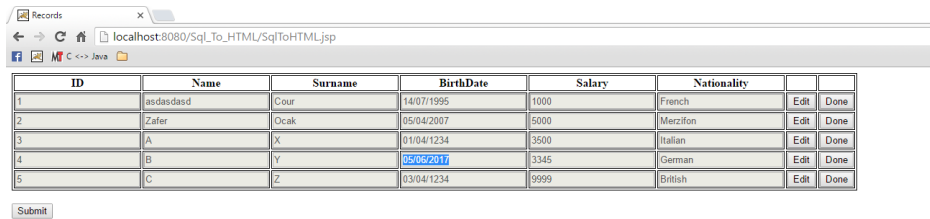
Second picture shows edit button's function on my web page:

Third picture shows done button's function on my web page:

The highligted boxes in the second and third pictures show the changes and functions of edit and done buttons.

Last picture is about the libraries has been used by my servlet and an image when the server started below:

- javax.servlet.ServletException,

- javax.servlet.annotation.WebServlet,

- javax.servlet.http.HttpServlet,

- javax.servlet.http.HttpServletRequest
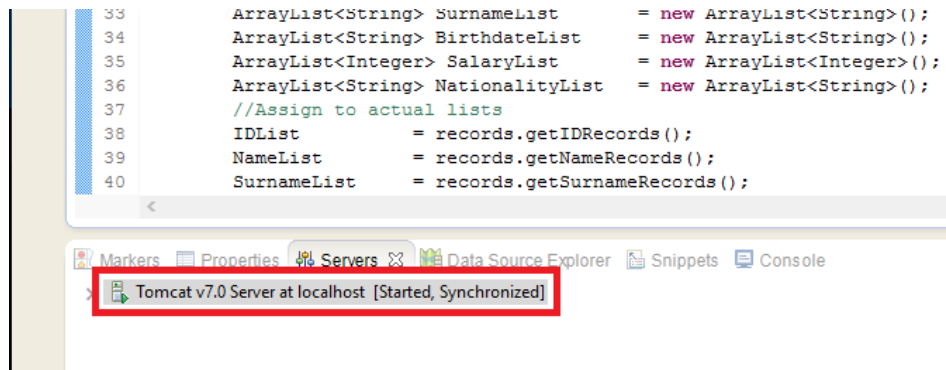
Overall, my intern has contributed plenty of things which I believe I am going to use in my future life. In my opinion, my intern showed me to see how time is important and how softwares make life easier and faster.