General Sir John Kotelawala Defence University

Faculty of Computing

Department of IT

Service Oriented Web Programming (IT 6023)

INSTRUCTOR ALLOCATION SYSTEM

LEVEL 6 - 2016

Student Declaration:

I declare that:

- *I understand what is meant by plagiarism*
- *The implications of plagiarism have been explained to me by my institution*
- *This assignment is all my own work and I have acknowledged any use of the published or unpublished works of other people.*

Student's signature: Date: 20.10.2016

Total number of pages including this cover page			63
Submission Date	20.10.2016	Due Date	20.10.2016
Student's ID	ICT / 14 / 051	Class Code	ICT_INTAKE XXXI
Student's Full Name	JKC Shayalika		
Lecturer's Name	Mr. TMKK Jinasena		

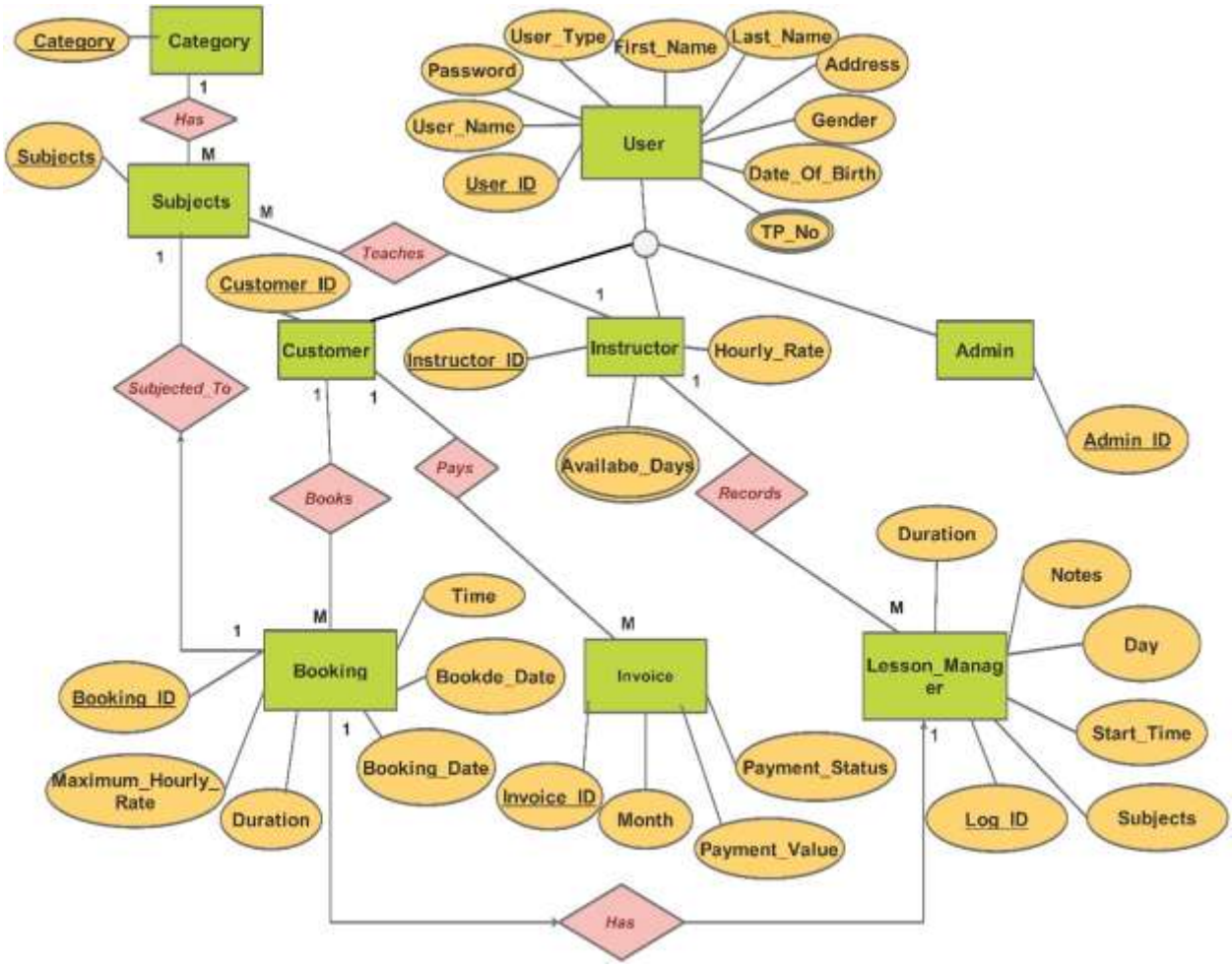
OFFICIAL USE ONLY

MARKER'S COMMENTS	
Marker's Name	Marks Awarded (100%)

Deliverables

1) Database

ER Diagram



Relational Mapping

1) ***Mapping regular entity types***

User (User_ID, User_Name, Password, First_Name, Last_Name, Address, Gender, DOB, TP_No)

- User (User_ID, User_Name, Password, First_Name, Last_Name, Address, Gender, DOB)
- User_TP (User_ID, TP_No)

Instructor (Instructor_ID, Available_Days, Hourly_Rate)

- Instructor (Instructor_ID, Hourly_Rate)
- Instructor_Available_Days (Instructor_ID, Available_Days)

2) ***Mapping regular entity types***

There are no weak entity types.

3) ***Mapping of Binary (one to one relationship)***

Booking (Booking_ID, Maximum_hourly_rate, Duration, Booking_Date, Time, Booked_Date)
Lesson_Manager (Log_ID, Start_Time, End_Time, Duration, Notes)

- Booking (Booking_ID, Maximum_hourly_rate, Duration, Booking_Date, Time, Booked_Date)
- Lesson_Manager (Log_ID, Start_Time, End_Time, Duration, Notes, Booking_ID(fk))

4) ***Mapping of Binary (1-M relationship)***

Instructor (Instructor_ID, Hourly_Rate)
Instructor_Available_Days (Instructor_ID, Available_Days)
Booking (Booking_ID, Maximum_hourly_rate, Duration, Booking_Date, Time, Booked_Date, Customer_ID(fk)))

- Booking (Booking_ID, Maximum_hourly_rate, Duration, Booking_Date, Time, Booked_Date, Instructor_ID(fk))

Instructor (Instructor_ID, Hourly_Rate)
Instructor_Available_Days (Instructor_ID, Available_Days)
Invoice (Invoice_ID, Month, Payment_Value, Payment_Status)

- Invoice (Invoice_ID, Month, Payment_Value, Payment_Status, Instructor_ID(fk))

Customer (Customer_ID)
Invoice (Invoice_ID, Month, Payment_Value, Payment_Status, Instructor_ID(fk))

- Lesson_Manager (Log_ID, Start_Time, End_Time, Duration, Notes, Booking_ID(fk), Instructor_ID(fk))

- Invoice (Invoice_ID, Month, Payment_Value, Payment_Status, Instructor_ID(fk), Customer_ID(fk))

Instructor (Instructor_ID, Hourly_Rate)
Instructor_Available_Days (Instructor_ID, Available_Days)
Lesson_Manager (Log_ID, Start_Time, End_Time, Duration, Notes, Booking_ID(fk))

Instructor (Instructor_ID, Hourly_Rate)
Instructor_Available_Days (Instructor_ID, Available_Days)
Subjects (Subjects)

- Subjects (Subjects, Instructor_ID(fk))

Subjects (Subjects, Instructor_ID(fk))
Category (Category)

- Subjects (Subjects, Instructor_ID(fk), Category(fk))

Subjects (Subjects, Instructor_ID(fk), Category(fk))
Booking (Booking_ID, Maximum_hourly_rate, Duration, Booking_Date, Time, Booked_Date, Instructor_ID(fk))

- Booking (Booking_ID, Maximum_hourly_rate, Duration, Booking_Date, Time, Booked_Date, Instructor_ID(fk), Subjects(fk))

5) ***Mapping of Binary (M-N relationship)***

There are no M-N relationships.

6) ***Mapping of Unary relationship***

There are no Unary relationships.

7) ***Mapping of Ternary (N-ary)***

There are no Ternary relationships.

8) ***Mapping of Super / Subtype relationship***

User (User_ID, User_Name, Password, First_Name, Last_Name, Address, Gender, DOB)
User_TP (User_ID, TP_No)
Customer (Customer_ID)
Instructor (Instructor_ID, Hourly_Rate)
Instructor_Available_Days (Instructor_ID, Available_Days)
Admin (Admin_ID)

- User (User_ID, User_Name, Password, First_Name, Last_Name, Address, Gender, DOB)

- User_TP (User_ID, TP_No)
- Customer (Customer_ID, User_ID)
- Instructor (Instructor_ID, Hourly_Rate, User_ID)
- Instructor_Available_Days (Instructor_ID, Available_Days)
- Admin (Admin_ID, User_ID)

Final Table Set

- User (User_ID, User_Name, Password, First_Name, Last_Name, Address, Gender, DOB)
- User_TP (User_ID, TP_No)
- Booking (Booking_ID, Maximum_hourly_rate, Duration, Booking_Date, Time, Booked_Date, Instructor_ID(fk), Subjects(fk))
- Invoice (Invoice_ID, Month, Payment_Value, Payment_Status, Instructor_ID(fk), Customer_ID(fk))
- Lesson_Manager (Log_ID, Start_Time, End_Time, Duration, Notes, Booking_ID(fk), Instructor_ID(fk))
- Customer (Customer_ID, User_ID)
- Instructor (Instructor_ID, Hourly_Rate, User_ID)
- Instructor_Available_Days (Instructor_ID, Available_Days)
- Admin (Admin_ID, User_ID)
- Category (Category)
- Subjects (Subjects, Instructor_ID(fk), Category(fk))

MySQL Database Query

```
-- phpMyAdmin SQL Dump
-- version 4.1.14
-- http://www.phpmyadmin.net
--
-- Host: 127.0.0.1
-- Generation Time: Oct 19, 2016 at 05:22 PM
-- Server version: 5.1.52-community
-- PHP Version: 5.5.12

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

--
-- Database: `sowpassignmentnew`
--

-- --------------------------------------------------------

--
-- Table structure for table `admin`
--

CREATE TABLE IF NOT EXISTS `admin` (
  `Admin_ID` int(11) NOT NULL AUTO_INCREMENT,
  `User_ID` int(11) NOT NULL,
  PRIMARY KEY (`Admin_ID`),
  KEY `User_ID` (`User_ID`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=3 ;

--
-- Dumping data for table `admin`
--

INSERT INTO `admin` (`Admin_ID`, `User_ID`) VALUES
(1, 4),
(2, 11);

-- --------------------------------------------------------

--
-- Table structure for table `booking`
--

CREATE TABLE IF NOT EXISTS `booking` (
  `Booking_ID` int(11) NOT NULL AUTO_INCREMENT,
  `Subjects` varchar(255) NOT NULL,
```

```
  `Booking_Date` datetime NOT NULL,
  `Booked_Date` datetime NOT NULL,
  `Maximum_Hourly_Rate` decimal(10,0) NOT NULL,
  `Start_Time` time NOT NULL,
  `Duration` decimal(10,0) NOT NULL,
  `CustomerID` int(11) NOT NULL,
  `Day` varchar(255) NOT NULL,
  PRIMARY KEY (`Booking_ID`),
  KEY `CustomerID` (`CustomerID`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=8 ;

--
-- Dumping data for table `booking`
--

INSERT INTO `booking` (`Booking_ID`, `Subjects`, `Booking_Date`,
`Booked_Date`, `Maximum_Hourly_Rate`, `Start_Time`, `Duration`, `CustomerID`,
`Day`) VALUES
(1, 'Chemistry', '2016-10-07 10:00:00', '2016-09-27 08:17:21', '788',
'08:00:00', '3', 1, 'Tuesday'),
(2, 'Chemistry', '2016-10-08 09:30:00', '2016-10-18 00:00:00', '150',
'11:00:00', '2', 2, 'Tuesday'),
(3, 'Chemistry', '2016-10-09 09:47:00', '2016-10-22 00:00:00', '200',
'18:00:00', '1', 1, 'Saturday'),
(7, 'Western Music', '2016-10-17 07:19:00', '2016-10-27 00:00:00', '300',
'12:00:00', '4', 1, 'Thursday');

-- --------------------------------------------------------

--
-- Table structure for table `category`
--

CREATE TABLE IF NOT EXISTS `category` (
  `Category` varchar(255) NOT NULL,
  PRIMARY KEY (`Category`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `category`
--

INSERT INTO `category` (`Category`) VALUES
('A/L'),
('Aesthetics'),
('Arts'),
('Commerce'),
('Management'),
('O/L'),
('Other'),
('Primary'),
('Technical Subjects');

-- --------------------------------------------------------
```

```
-- Table structure for table `customer`
--

CREATE TABLE IF NOT EXISTS `customer` (
  `Customer_ID` int(11) NOT NULL AUTO_INCREMENT,
  `User_ID` int(11) NOT NULL,
  PRIMARY KEY (`Customer_ID`),
  KEY `User_ID` (`User_ID`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=5 ;

--
-- Dumping data for table `customer`
--

INSERT INTO `customer` (`Customer_ID`, `User_ID`) VALUES
(1, 2),
(2, 3),
(3, 9),
(4, 10);

-- --------------------------------------------------------

--
-- Table structure for table `instructor`
--

CREATE TABLE IF NOT EXISTS `instructor` (
  `Instructor_ID` int(11) NOT NULL AUTO_INCREMENT,
  `User_ID` int(255) NOT NULL,
  `Hourly_Rate` decimal(10,0) NOT NULL,
  PRIMARY KEY (`Instructor_ID`),
  KEY `User_ID` (`User_ID`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;

--
-- Dumping data for table `instructor`
--

INSERT INTO `instructor` (`Instructor_ID`, `User_ID`, `Hourly_Rate`) VALUES
(1, 1, '2000'),
(2, 5, '2000'),
(3, 6, '300'),
(4, 7, '200'),
(5, 8, '200');

-- --------------------------------------------------------

--
-- Table structure for table `instructor_available_days`
--

CREATE TABLE IF NOT EXISTS `instructor_available_days` (
  `Instructor_ID` int(11) NOT NULL AUTO_INCREMENT,
  `Available_Days` varchar(255) NOT NULL,
  PRIMARY KEY (`Instructor_ID`)
```

```
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;

--
-- Dumping data for table `instructor_available_days`
--

INSERT INTO `instructor_available_days` (`Instructor_ID`, `Available_Days`)
VALUES
(1, 'Tuesday,Wednesday'),
(2, 'Tuesday'),
(3, 'Thursday'),
(4, 'Wednesday'),
(5, 'Wednesday');

-- --------------------------------------------------------

--
-- Table structure for table `invoice`
--

CREATE TABLE IF NOT EXISTS `invoice` (
  `Invoice_ID` int(11) NOT NULL AUTO_INCREMENT,
  `Customer_ID` int(11) NOT NULL,
  `Month` varchar(255) NOT NULL,
  `Payment_Value` decimal(10,0) NOT NULL,
  `Payment_Status` varchar(100) NOT NULL,
  PRIMARY KEY (`Invoice_ID`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=4 ;

--
-- Dumping data for table `invoice`
--

INSERT INTO `invoice` (`Invoice_ID`, `Customer_ID`, `Month`, `Payment_Value`,
`Payment_Status`) VALUES
(1, 1, 'August', '1000', 'Paid'),
(2, 2, 'August', '2000', 'Paid'),
(3, 1, 'September', '1250', 'Not Paid');

-- --------------------------------------------------------

--
-- Table structure for table `lessonmanager`
--

CREATE TABLE IF NOT EXISTS `lessonmanager` (
  `Log_ID` int(11) NOT NULL,
  `Start_Time` varchar(200) NOT NULL,
  `Duration` varchar(200) NOT NULL,
  `Notes` varchar(255) NOT NULL,
  `Instructor_ID` varchar(255) NOT NULL,
  PRIMARY KEY (`Log_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
```

```
-- Dumping data for table `lessonmanager`
--

INSERT INTO `lessonmanager` (`Log_ID`, `Start_Time`, `Duration`, `Notes`,
`Instructor_ID`) VALUES
(1, '4', '1', 'Conducted Successfully', '1'),
(2, '3', '2', 'Conducted Successfully', '2'),
(3, '7', '11', 'Conducted', '3'),
(4, '3', '3', 'Success', '1'),
(8, '5', '2', 'Not good', '5'),
(9, '3', '1', 'Successfull', '1'),
(10, '7', '2', 'Successfull lecture', '1'),
(11, '6', '1', 'null', '3');

-- --------------------------------------------------------

--
-- Table structure for table `lessonmanagerapp`
--

CREATE TABLE IF NOT EXISTS `lessonmanagerapp` (
  `Log_ID` int(11) NOT NULL AUTO_INCREMENT,
  `Booking_ID` int(11) NOT NULL DEFAULT '0',
  `Start_Time` time NOT NULL DEFAULT '00:00:00',
  `Duration` decimal(10,0) NOT NULL DEFAULT '0',
  `Day` varchar(100) NOT NULL DEFAULT '0',
  `Subject` varchar(100) NOT NULL DEFAULT '0',
  `Instructor_ID` int(11) NOT NULL DEFAULT '0',
  `Note` varchar(255) NOT NULL DEFAULT '0',
  PRIMARY KEY (`Log_ID`),
  KEY `Booking_ID` (`Booking_ID`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=8 ;

--
-- Dumping data for table `lessonmanagerapp`
--

INSERT INTO `lessonmanagerapp` (`Log_ID`, `Booking_ID`, `Start_Time`,
`Duration`, `Day`, `Subject`, `Instructor_ID`, `Note`) VALUES
(1, 0, '00:00:00', '0', '0', '0', 2, '0'),
(2, 0, '00:00:00', '0', '0', '0', 3, '0'),
(4, 1, '08:00:00', '3', 'Tuesday', 'Chemistry', 0, '0'),
(5, 2, '11:00:00', '2', 'Tuesday', 'Chemistry', 0, '0'),
(6, 3, '18:00:00', '1', 'Saturday', 'Chemistry', 0, '0'),
(7, 7, '12:00:00', '4', 'Thursday', 'Western Music', 0, '0');

-- --------------------------------------------------------

--
-- Table structure for table `schedule`
--

CREATE TABLE IF NOT EXISTS `schedule` (
  `Schedule_ID` int(11) NOT NULL AUTO_INCREMENT,
  `Booking_ID` int(11) NOT NULL,
```

```
  `Instructor_ID` int(11) NOT NULL,
  `Subject` varchar(255) NOT NULL,
  `Booked_Date` datetime NOT NULL,
  `Start_Time` time NOT NULL,
  `Duration` decimal(10,0) NOT NULL,
  `Customer_ID` int(11) NOT NULL,
  `Day` varchar(255) NOT NULL,
  PRIMARY KEY (`Schedule_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

-- --------------------------------------------------------

--
-- Table structure for table `subjectcategory`
--

CREATE TABLE IF NOT EXISTS `subjectcategory` (
  `Subjects` varchar(100) NOT NULL,
  `Category` varchar(100) NOT NULL,
  PRIMARY KEY (`Subjects`,`Category`),
  KEY `Subjects` (`Subjects`),
  KEY `Category` (`Category`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `subjectcategory`
--

INSERT INTO `subjectcategory` (`Subjects`, `Category`) VALUES
('BC', 'Arts'),
('Biology', 'A/L'),
('Business Studies', 'Management'),
('Chemistry', 'A/L'),
('Combined Maths', 'A/L'),
('Eastern Music', 'Aesthetics'),
('Scolarship', 'Primary'),
('Sinhala', 'Arts'),
('Western Music', 'Aesthetics');

-- --------------------------------------------------------

--
-- Table structure for table `subjects`
--

CREATE TABLE IF NOT EXISTS `subjects` (
  `Subjects` varchar(255) NOT NULL,
  `Instructor_ID` int(11) NOT NULL AUTO_INCREMENT,
  `Category` varchar(255) NOT NULL,
  PRIMARY KEY (`Subjects`,`Instructor_ID`),
  KEY `Instructor_ID` (`Instructor_ID`,`Category`),
  KEY `Category` (`Category`),
  KEY `Instructor_ID_2` (`Instructor_ID`),
  KEY `Subjects` (`Subjects`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;
```

```
--
-- Dumping data for table `subjects`
--

INSERT INTO `subjects` (`Subjects`, `Instructor_ID`, `Category`) VALUES
('Biology', 5, 'A/L'),
('Chemistry', 2, 'A/L'),
('Eastern Music,Western Music', 1, 'Aesthetics'),
('Western Music', 3, 'Aesthetics'),
('BC', 4, 'Arts');

-- --------------------------------------------------------

--
-- Table structure for table `user`
--
CREATE TABLE IF NOT EXISTS `user` (
  `User_ID` int(11) NOT NULL AUTO_INCREMENT,
  `User_Name` varchar(100) NOT NULL,
  `Password` varchar(100) NOT NULL,
  `User_Type` varchar(100) NOT NULL,
  `First_Name` varchar(100) NOT NULL,
  `Last_Name` varchar(100) NOT NULL,
  `Address` varchar(100) NOT NULL,
  `Gender` varchar(100) NOT NULL,
  `Date_Of_Birth` date NOT NULL,
  `TP_No` varchar(100) NOT NULL,
  PRIMARY KEY (`User_ID`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=12 ;

--
-- Dumping data for table `user`
--

INSERT INTO `user` (`User_ID`, `User_Name`, `Password`, `User_Type`,
`First_Name`, `Last_Name`, `Address`, `Gender`, `Date_Of_Birth`, `TP_No`)
VALUES
(1, 'chathurangijks@gmail.com', 'sc', 'Instructor', 'Chathurangi', 'Shyalika',
'No,57, Madelgamuwa, Gampaha', 'Female', '1998-02-02', '0772150269'),
(2, 'chamanijks2@gmail.com', 'cd', 'Customer', 'Chamani', 'Shiranthika',
'No56, Silva Rd, Moratuwa', 'Female', '1008-02-02', '0111234561'),
(3, 'jayakodydw@gmail.com', 's', 'Customer', 'Jaya', 'Sampath',
'No56,289Road', 'Male', '9988-02-09', '0774567891'),
(4, 'dwjayakody@gmail.com', 'd', 'Administrator', 'Dayawansa', 'Jayakody',
'No56, Silva Rd, Moratuwa', 'Male', '1980-12-02', '0779886575'),
(5, 'd@gmail.com', 'da', 'Instructor', 'Damith', 'Sumathipala', 'No 45,
Nedagamuwa, Gampaha', 'Male', '1978-02-11', '0774567891'),
(6, 'pradeepa@yahoo.com', 'mw', 'Instructor', 'Pradeepa', 'Satharasinghe',
'No.45, Asgiriya, Gampaha', 'Female', '2015-12-10', '0332230981'),
(7, 'sagara@gmail.com', 'pw', 'Instructor', 'Sagara', 'Palansuriya', 'No.5,
Kandy Rd', 'Female', '2016-10-18', '0772150260'),
(8, 'dil@gmail.com', 'k', 'Instructor', 'Dilruwan', 'Senadeera', 'No.56,
Palama Rd, Asgiriya', 'Female', '2016-10-11', '0772150234'),
```

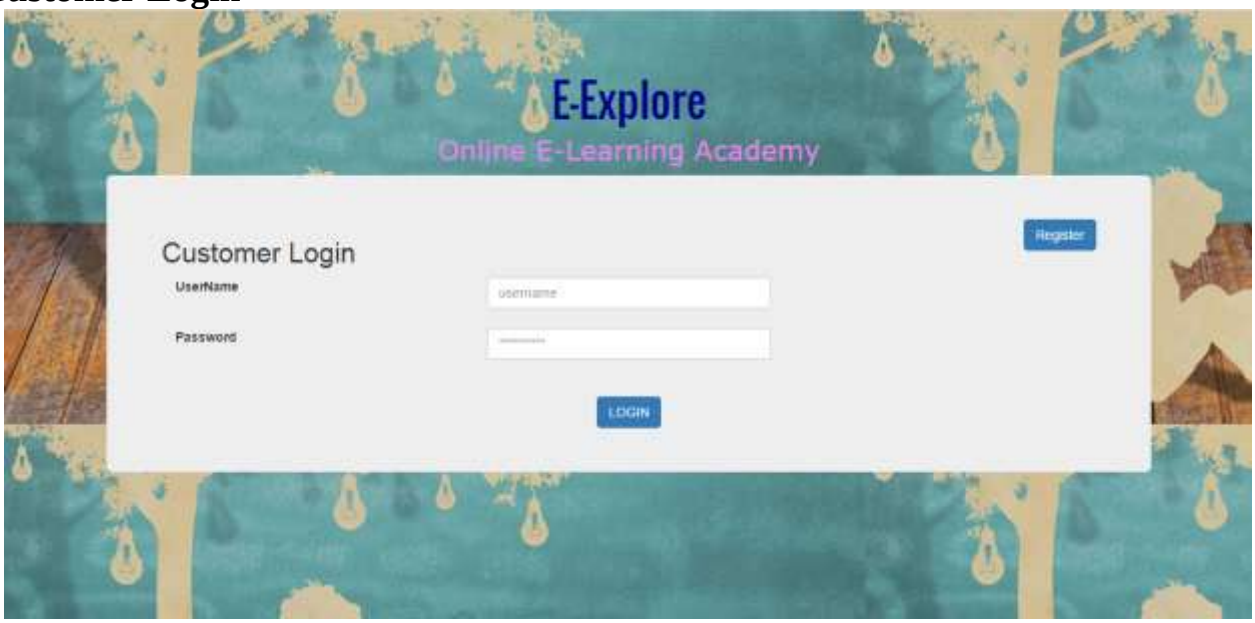
```
(9, 'gimi@yahoo.com', 'gm', 'Customer', 'Gimhani', 'Hangawaththa', 'No.23
Samagi Mw, Piliyandala', 'Female', '2016-10-10', '0725037410'),
(10, 'sri@gmail.com', 's', 'Customer', 'Srimali', 'Manchanayaka', 'No 89,
Ihala Imbulgoda', 'Female', '2016-10-05', '0725048961'),
(11, 'kamal@hotmail.com', 'km', 'Administrator', 'Kamal', 'Jayakody', 'No.
23/A, Panadura', 'Male', '2016-10-10', '0779883572');

-- --------------------------------------------------------

--
-- Table structure for table `user_tp`
--

CREATE TABLE IF NOT EXISTS `user_tp` (
  `User_ID` int(11) NOT NULL AUTO_INCREMENT,
  `TP_No` varchar(100) NOT NULL,
  PRIMARY KEY (`User_ID`,`TP_No`),
  KEY `TP_No` (`TP_No`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

--
-- Constraints for dumped tables
--

--
-- Constraints for table `admin`
--
ALTER TABLE `admin`
  ADD CONSTRAINT `admin_ibfk_1` FOREIGN KEY (`User_ID`) REFERENCES `user`
(`User_ID`);

--
-- Constraints for table `customer`
--
ALTER TABLE `customer`
  ADD CONSTRAINT `customer_ibfk_1` FOREIGN KEY (`User_ID`) REFERENCES `user`
(`User_ID`);

--
-- Constraints for table `instructor`
--
ALTER TABLE `instructor`
  ADD CONSTRAINT `instructor_ibfk_1` FOREIGN KEY (`User_ID`) REFERENCES `user`
(`User_ID`);

--
-- Constraints for table `instructor_available_days`
--
ALTER TABLE `instructor_available_days`
  ADD CONSTRAINT `instructor_available_days_ibfk_1` FOREIGN KEY
(`Instructor_ID`) REFERENCES `instructor` (`Instructor_ID`);

--
-- Constraints for table `subjectcategory`
--
```

```
ALTER TABLE `subjectcategory`
  ADD CONSTRAINT `subjectcategory_ibfk_1` FOREIGN KEY (`Category`) REFERENCES
`category` (`Category`);

--
-- Constraints for table `subjects`
--
ALTER TABLE `subjects`
  ADD CONSTRAINT `subjects_ibfk_1` FOREIGN KEY (`Instructor_ID`) REFERENCES
`instructor` (`Instructor_ID`),
  ADD CONSTRAINT `subjects_ibfk_2` FOREIGN KEY (`Category`) REFERENCES
`category` (`Category`);

--
-- Constraints for table `user_tp`
--
ALTER TABLE `user_tp`
  ADD CONSTRAINT `user_tp_ibfk_1` FOREIGN KEY (`User_ID`) REFERENCES `user`
(`User_ID`);

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

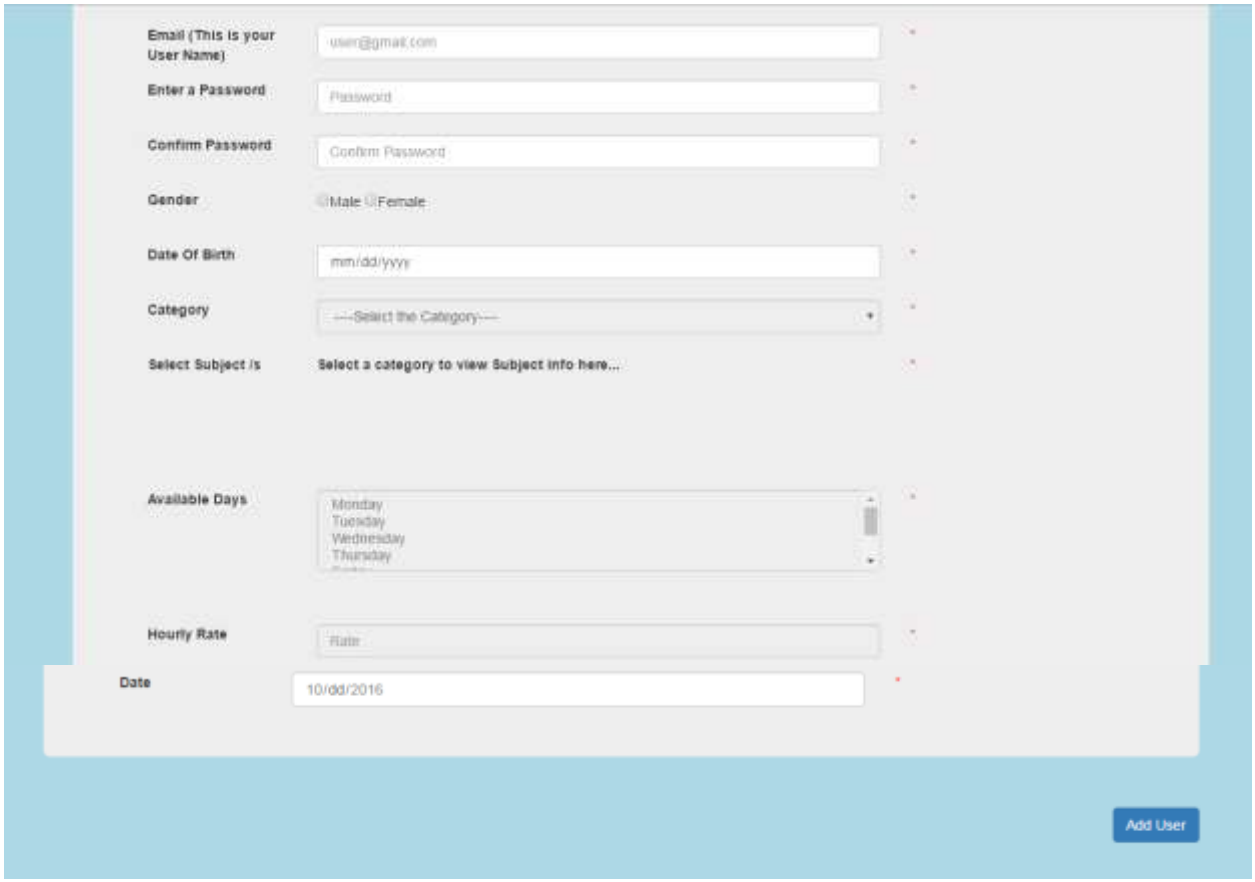
2) Web Application

Customer Interfaces

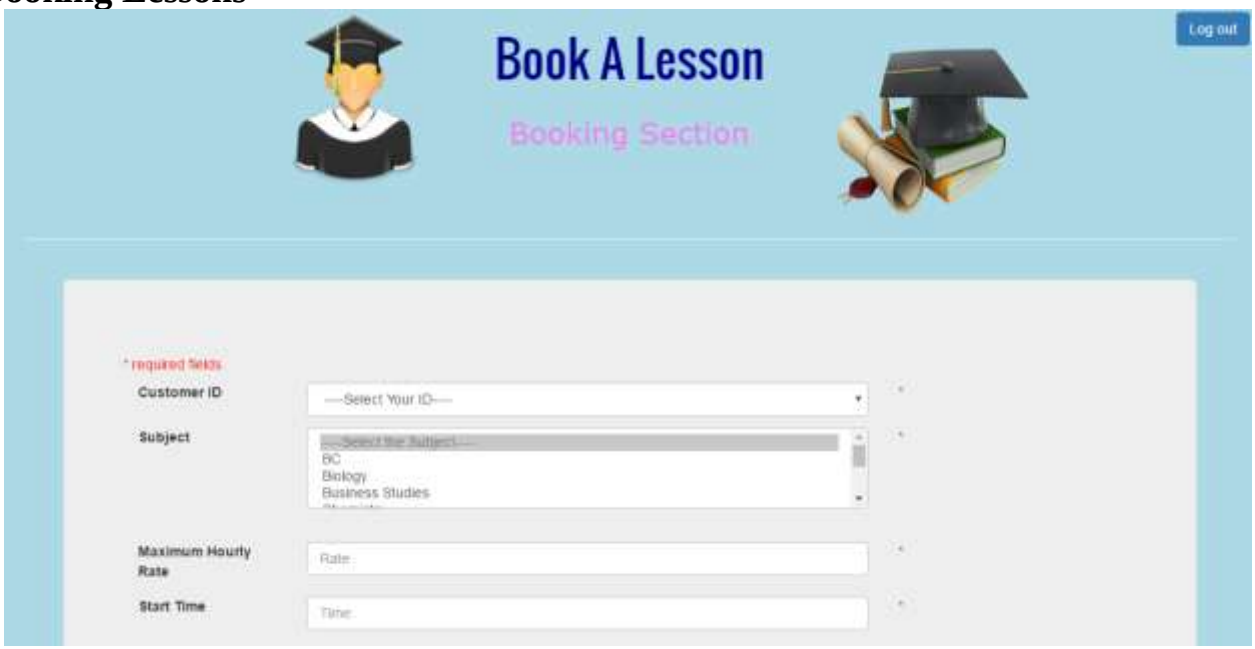
Customer Login

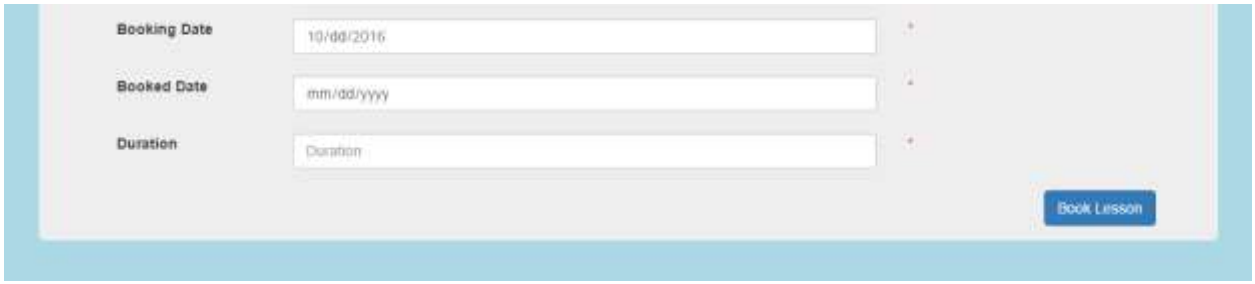


Registering a new Customer



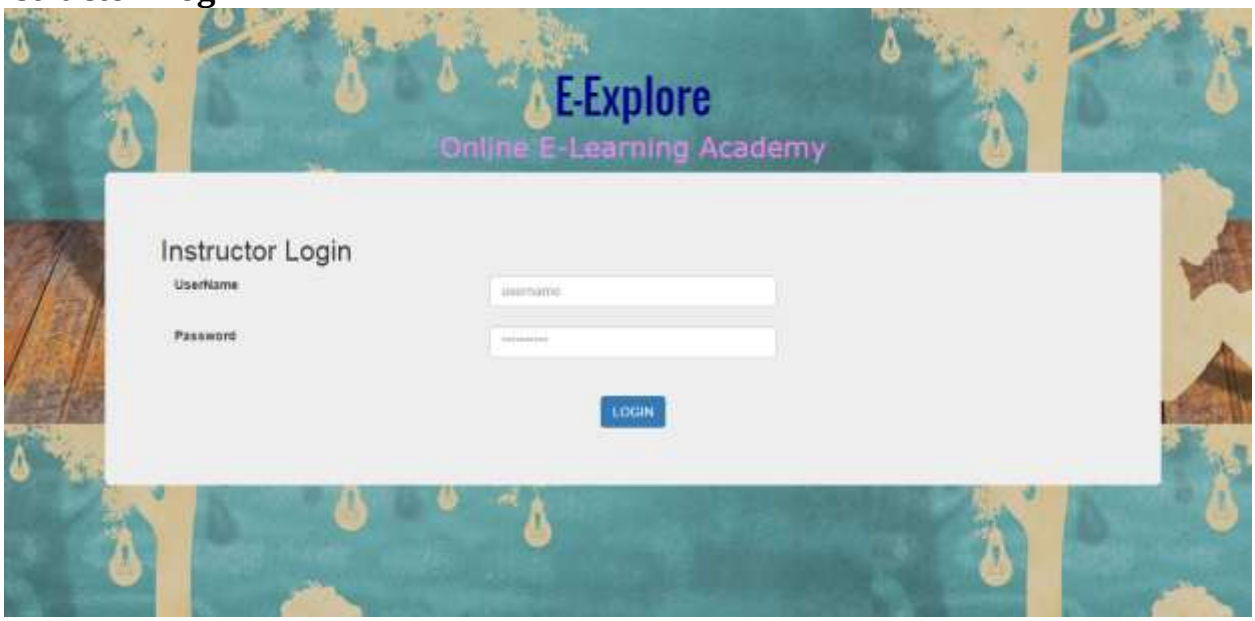
Booking Lessons



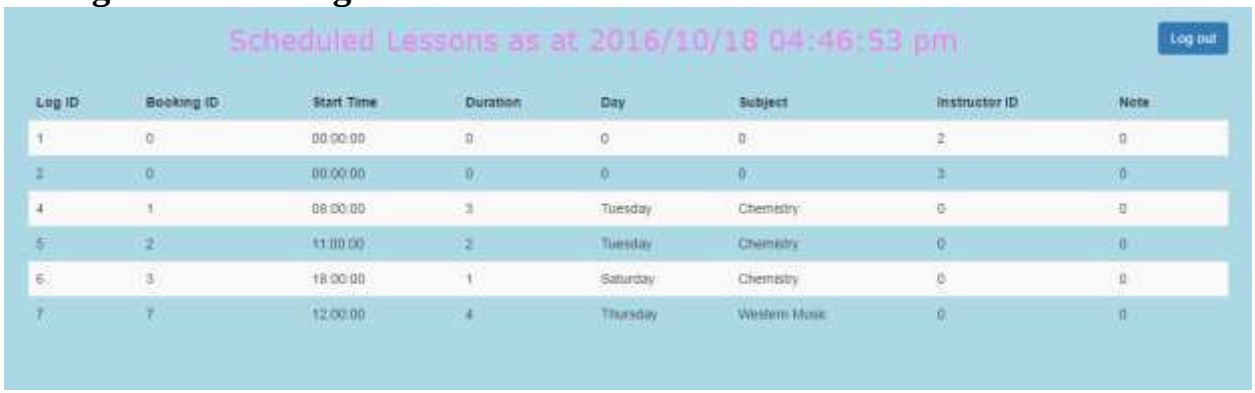


Instructor Interfaces

Instructor Login



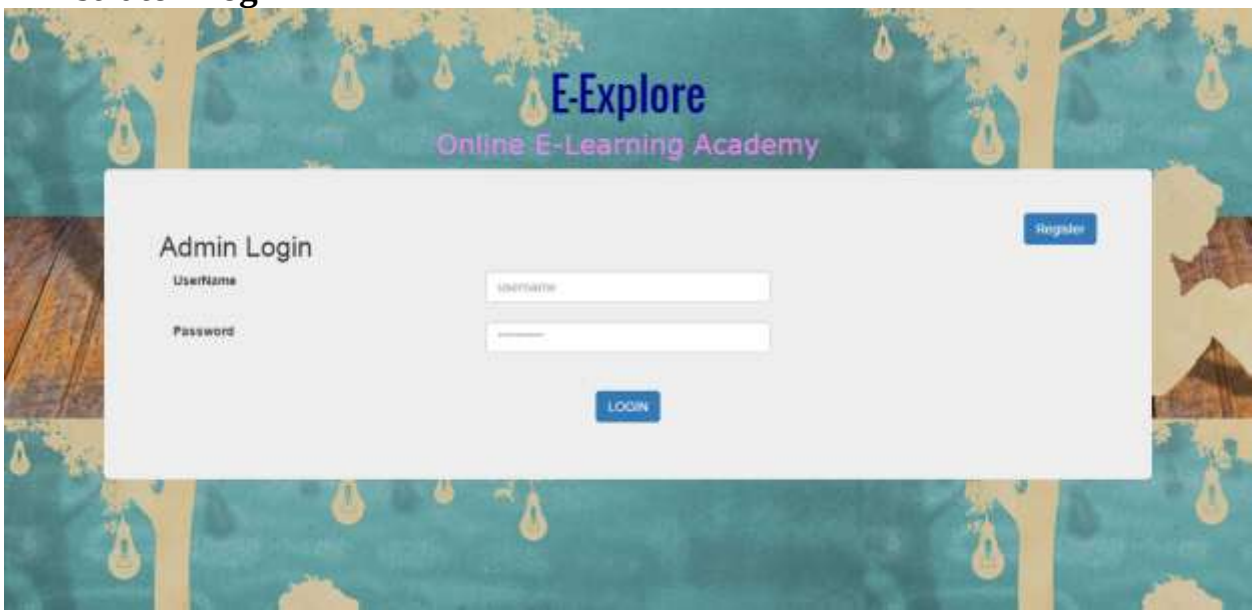
Viewing Lesson Manager Details



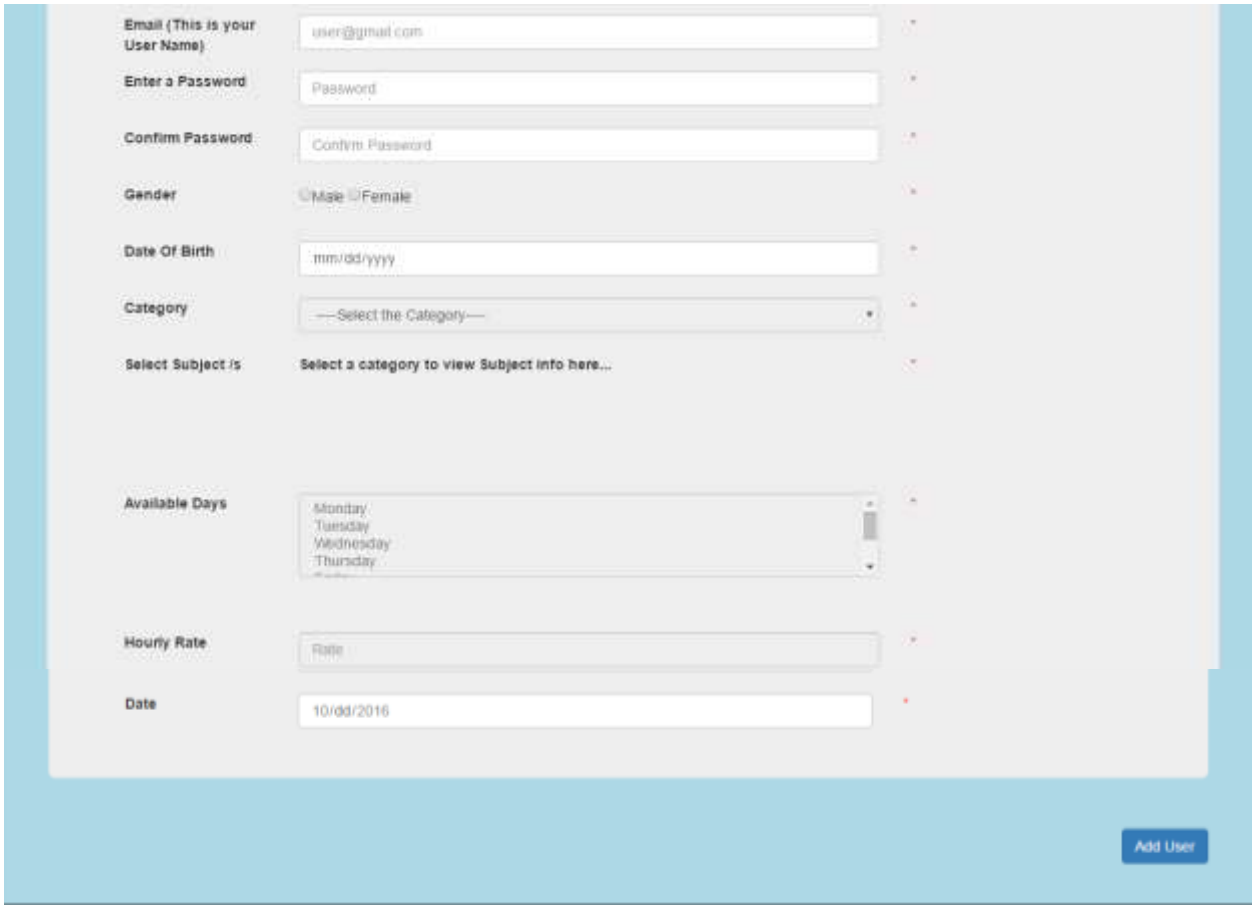
Log ID	Booking ID	Start Time	Duration	Day	Subject	Instructor ID	Note
1	0	00:00:00	0	0	0	2	0
2	0	00:00:00	0	0	0	3	0
4	1	08:00:00	3	Tuesday	Chemistry	0	0
5	2	11:00:00	2	Tuesday	Chemistry	0	0
6	3	18:00:00	1	Saturday	Chemistry	0	0
7	7	12:00:00	4	Thursday	Western Music	0	0

Admin Interfaces

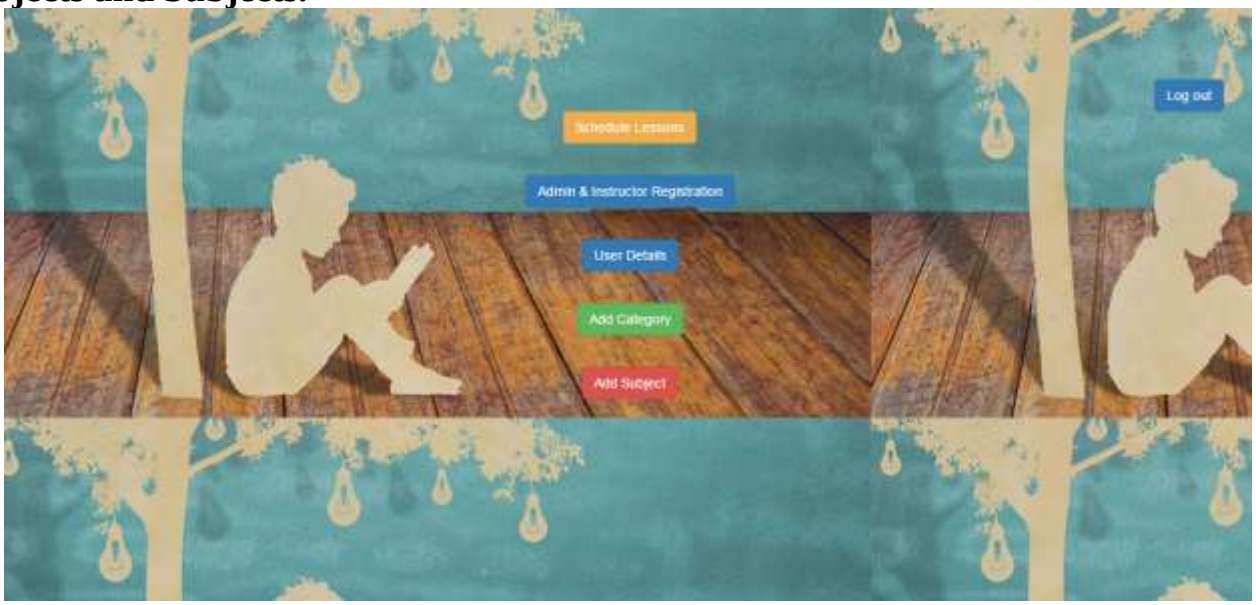
Administrator Login



Registering of Another administrators



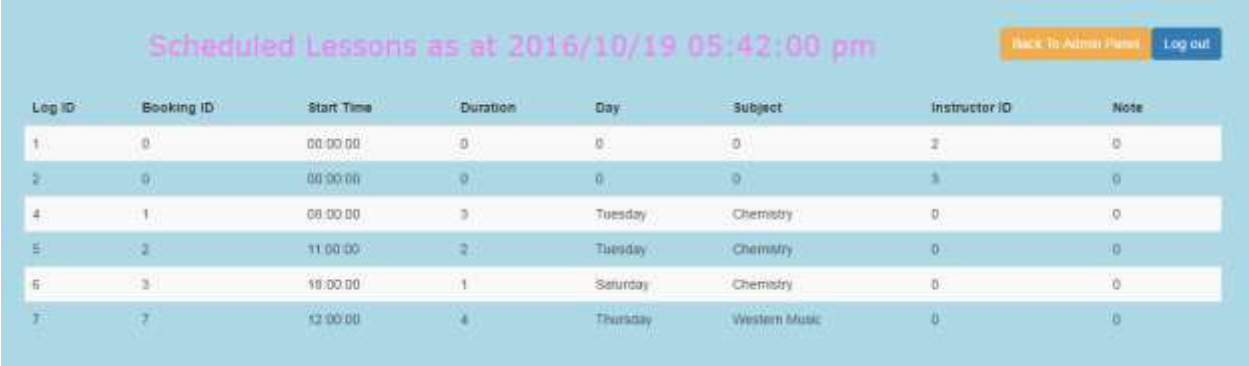
Admin Panel- Admin can schedule lessons, Register Instructors and another administrators for the system, View, Update User details, Add Categories of subjects and Subjects.



Scheduling Lessons by the Admin

Success message after scheduling lessons for a given week

View Scheduled Lessons

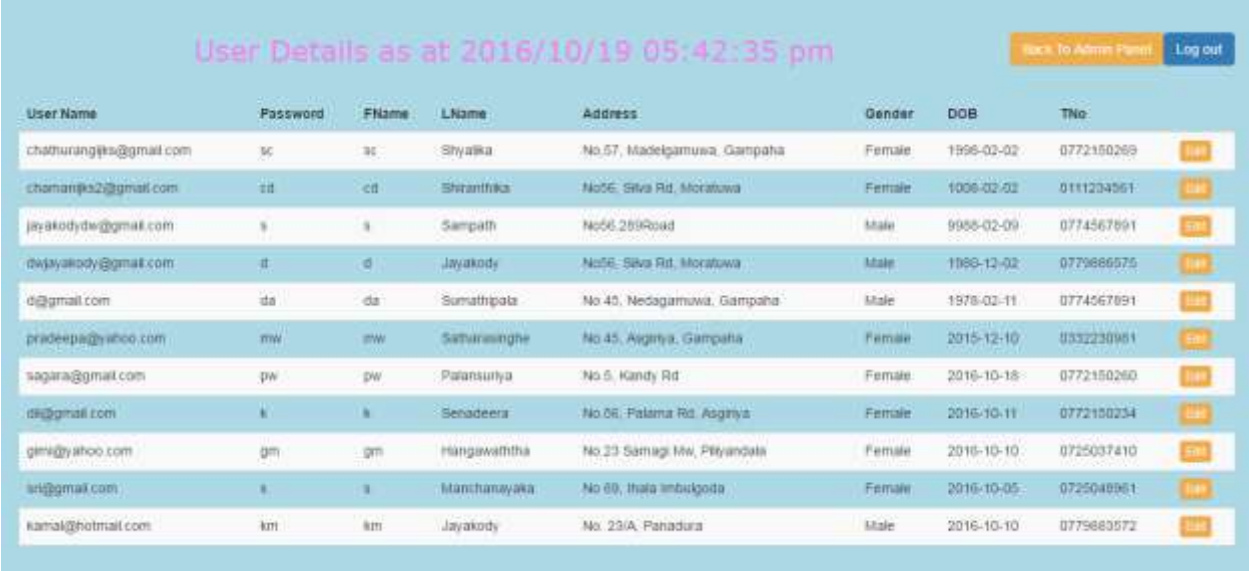


Scheduled Lessons as at 2016/10/19 05:42:00 pm

Back To Admin Panel | Log out

Log ID	Booking ID	Start Time	Duration	Day	Subject	Instructor ID	Note
1	0	00:00:00	0	0	0	2	0
2	0	00:00:00	0	0	0	3	0
4	1	08:00:00	3	Tuesday	Chemistry	0	0
5	2	11:00:00	2	Tuesday	Chemistry	0	0
6	3	10:00:00	1	Saturday	Chemistry	0	0
7	7	12:00:00	4	Thursday	Western Music	0	0

View, Update User details



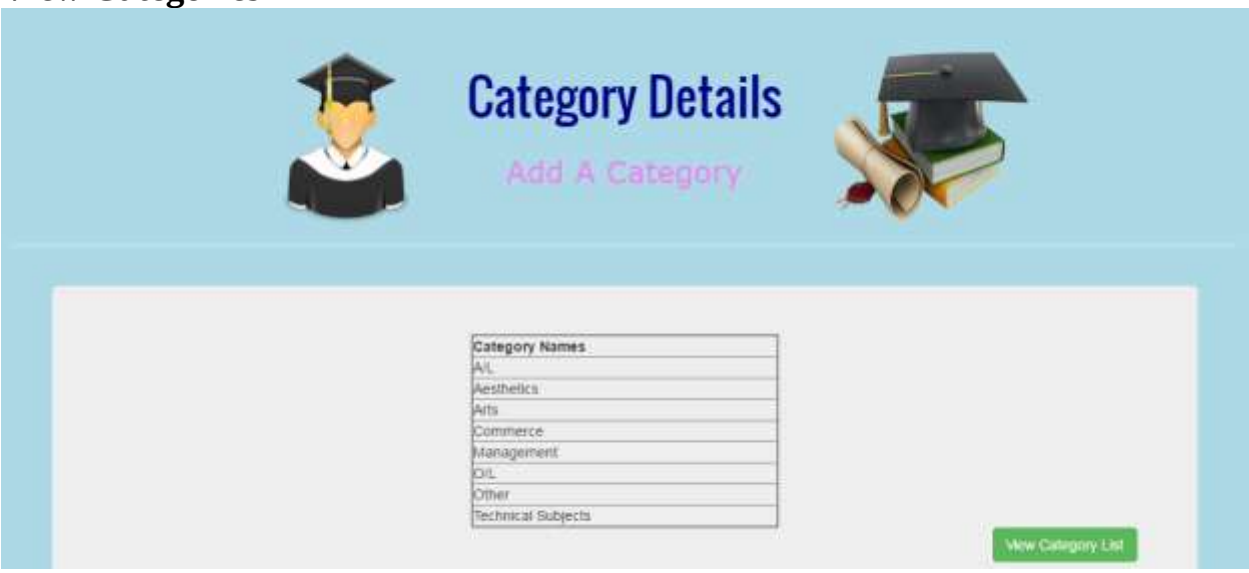
User Details as at 2016/10/19 05:42:35 pm

Back To Admin Panel | Log out

User Name	Password	FName	LName	Address	Gender	DOB	TNo	
chathurangiks@gmail.com	sc	sc	Shyalika	No.57, Madelgamuwa, Gampaha	Female	1996-02-02	0772150269	Edit
chamanjks2@gmail.com	cd	cd	Shiranthika	No56, Silva Rd, Moratuwa	Female	1006-02-02	0111234561	Edit
jayakodydw@gmail.com	s	s	Sampath	No56,289Road	Male	9966-02-09	0774567891	Edit
dwjayakody@gmail.com	d	d	Jayakody	No56, Silva Rd, Moratuwa	Male	1980-12-02	0779886575	Edit
d@gmail.com	da	da	Sumathipala	No 45, Nedagamuwa, Gampaha	Male	1978-02-11	0774567891	Edit
pradeepa@yahoo.com	mw	mw	Sathurasinghe	No 45, Aigiriya, Gampaha	Female	2015-12-10	0332230981	Edit
sagara@gmail.com	pw	pw	Palansuriya	No 5, Kandy Rd	Female	2016-10-18	0772150260	Edit
dk@gmail.com	k	k	Senadeera	No.06, Palama Rd, Asgiriya	Female	2016-10-11	0772150234	Edit
gimi@yahoo.com	gm	gm	Hangawaththa	No.23 Samagi Mw, Piliyandala	Female	2016-10-10	0725037410	Edit
sri@gmail.com	s	s	Manchanayaka	No 69, Ihala Imbulgoda	Female	2016-10-05	0725048961	Edit
kamal@hotmail.com	km	km	Jayakody	No. 23/A, Panadura	Male	2016-10-10	0779883572	Edit

Add Categories

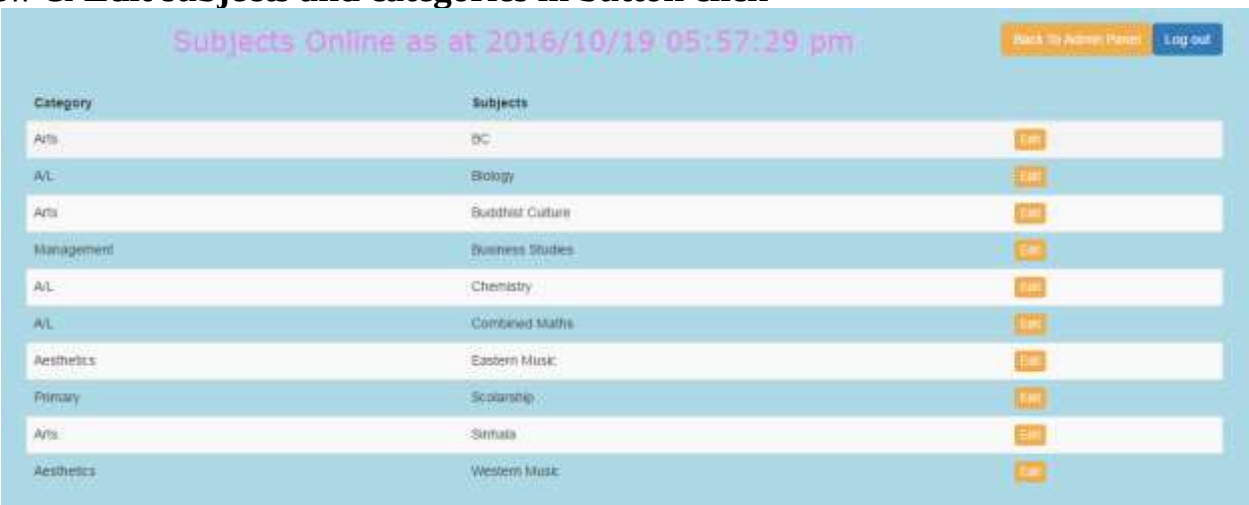
View Categories



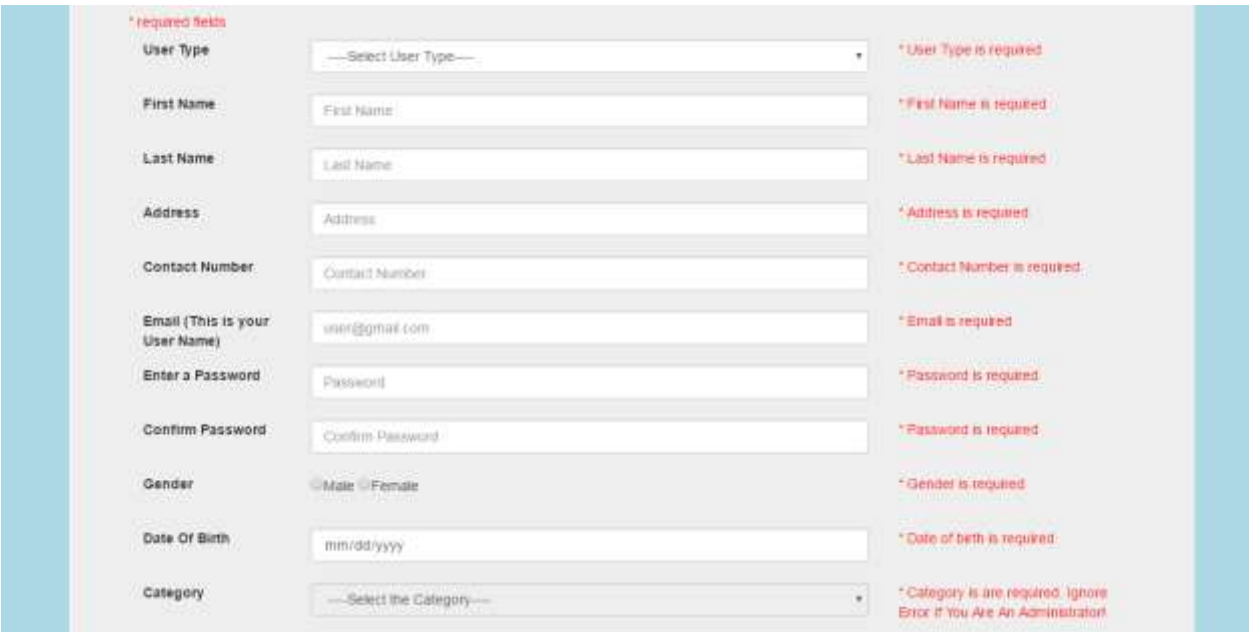
Add Subjects

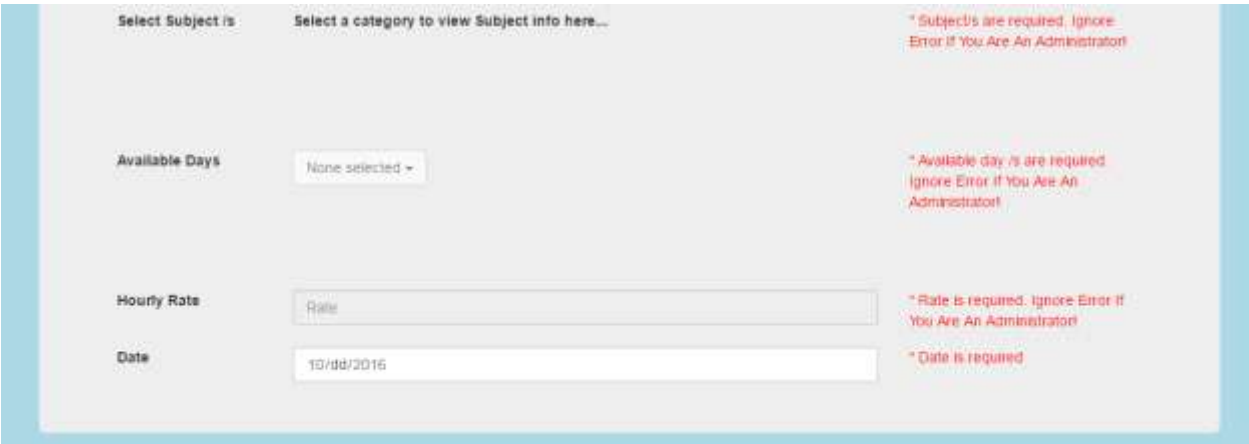
View Subjects

View & Edit subjects and categories in button click

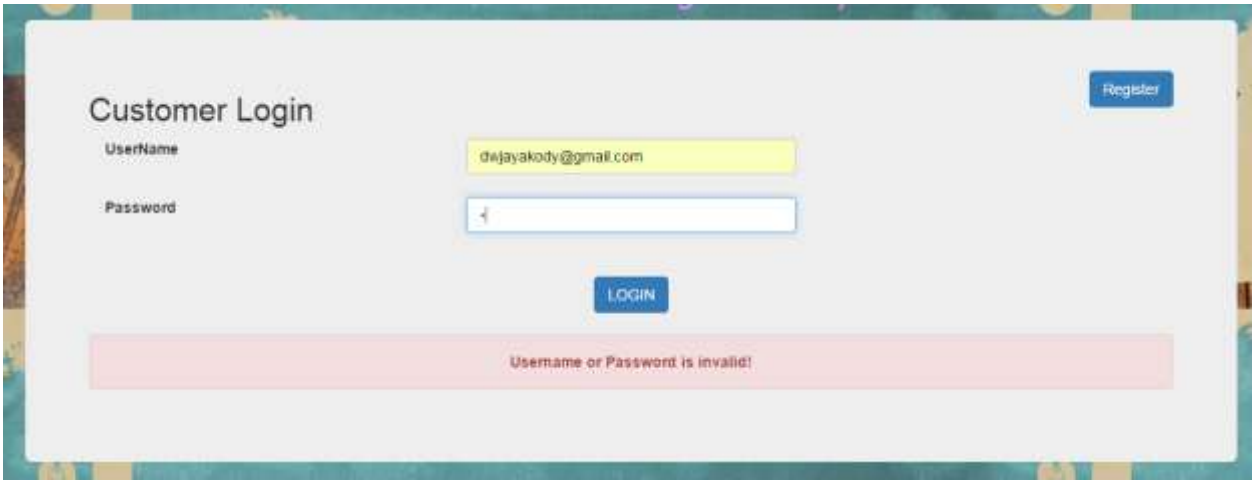


Data Validation





User Authentication



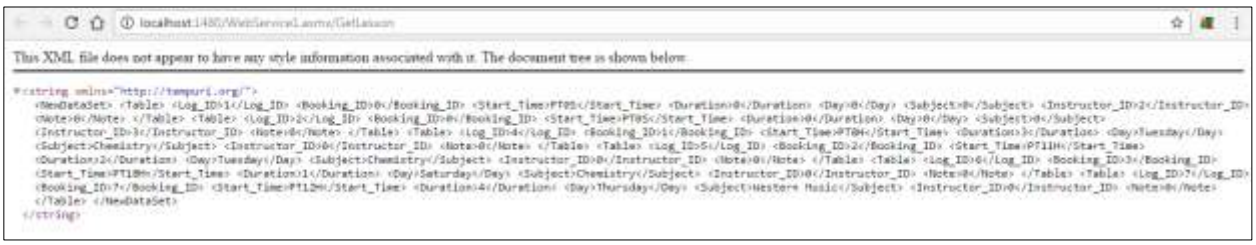
3) SOAP Web Service

Methods in WebService1.asmx

```
[WebMethod]
        public string GetLesson(int Instructor_ID)
        {
            String sql = "Select * from lessonmanagerapp";
            DataSet dataset = conObj.execQuery(sql);
            StringWriter sw = new StringWriter();
            dataset.WriteXml(sw, XmlWriteMode.IgnoreSchema);
            string xmlResult = sw.ToString();
            return xmlResult;

        }

        [WebMethod]
        public string GetLessonMgrDetails(int Booking_ID)
        {
            String sql = "Select * from lessonmanagerapp where Booking_ID ='" +
Booking_ID + "' ";
            DataSet dataset = conObj.execQuery(sql);
            StringWriter sw = new StringWriter();
            dataset.WriteXml(sw, XmlWriteMode.IgnoreSchema);
            string xmlResult = sw.ToString();
            return xmlResult;

        }

        [WebMethod]
        public string UploadNotes(string Start_Time, decimal duration, string note)
        {
            string connetionString = null;
            MySqlConnection connection;
            MySqlCommand command;
            MySqlDataAdapter adpter = new MySqlDataAdapter();
            DataSet ds = new DataSet();
            XmlReader xmlFile;
            string sql = null;
            string filePath = "";

            string StartTime = "";
            string Duration = "";
            string Note = "";
            string InstructorID = "";

            connetionString = "Database=sowpassignmentnew; Data Source=localhost; User
Id=root; Password=123";

            connection = new MySqlConnection(connetionString);
            connection.Open();


            xmlFile = XmlReader.Create("D:\\LessonManagerApp.xml", new
XmlReaderSettings());
            ds.ReadXml(xmlFile);
```

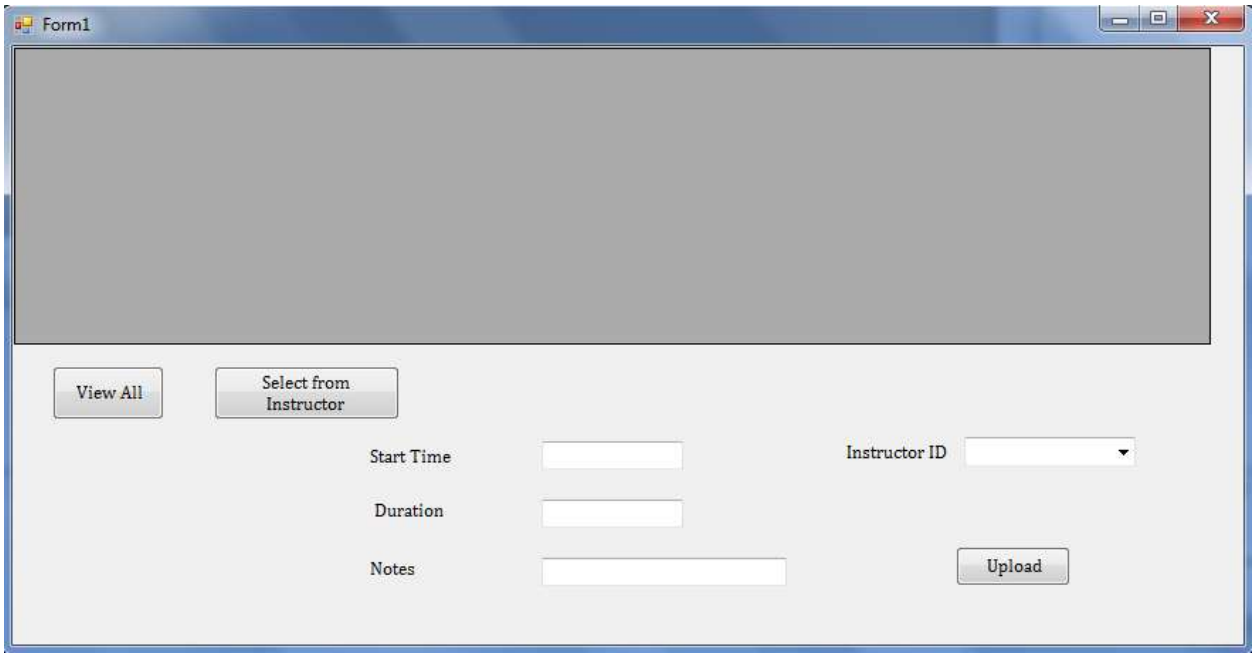
```
            int i = 0;

            XmlDocument xml = new XmlDocument();
            filePath = @"D:\\LessonManagerApp.xml";
            xml.Load(filePath);

            int count = xml.SelectNodes("Users/User").Count;


            StartTime = xml.SelectSingleNode("Users/User[last()]/StartTime").InnerText;
            Duration = xml.SelectSingleNode("Users/User[last()]/Duration").InnerText;
            Note = xml.SelectSingleNode("Users/User[last()]/Notes").InnerText;
            InstructorID =
xml.SelectSingleNode("Users/User[last()]/Instructor_ID").InnerText;


            sql = "insert into lessonmanager values('" + (count) + "','" + StartTime +
"','" + Duration + "','" + Note + "','" + InstructorID + "')";


            command = new MySqlCommand(sql, connection);
            adpter.InsertCommand = command;
            adpter.InsertCommand.ExecuteNonQuery();
            connection.Close();

            StringWriter sw = new StringWriter();
            string xmlResult = sw.ToString();
            return xmlResult;
        }
```

Result of GetLesson method

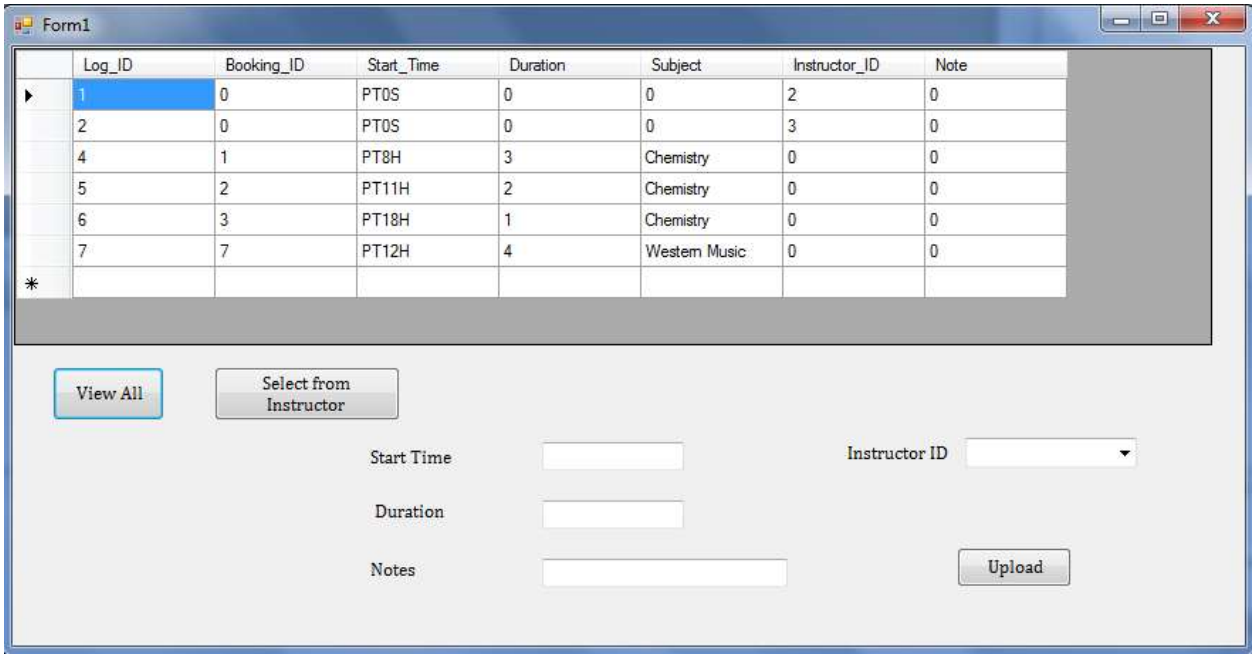


Result of GetLessonMgrDetails-
Returns the booking details according to the booking id.

4) Lesson Manager Application for Instructors

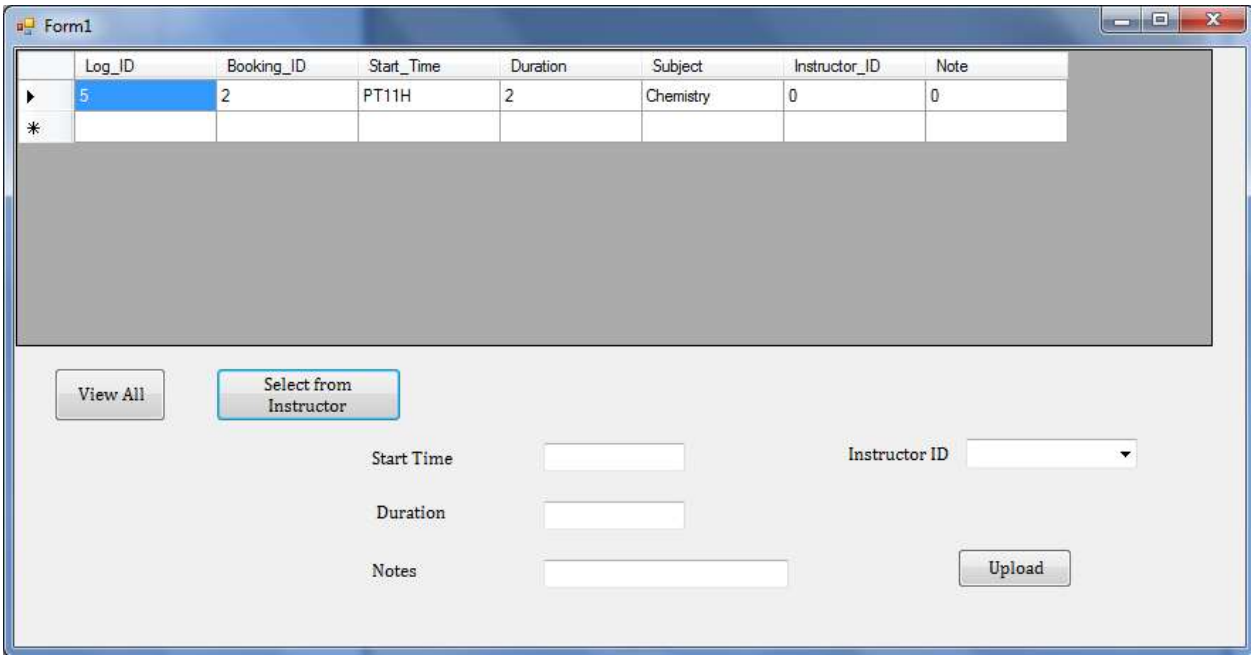


"View All" Button- returns all the lessons booked so far

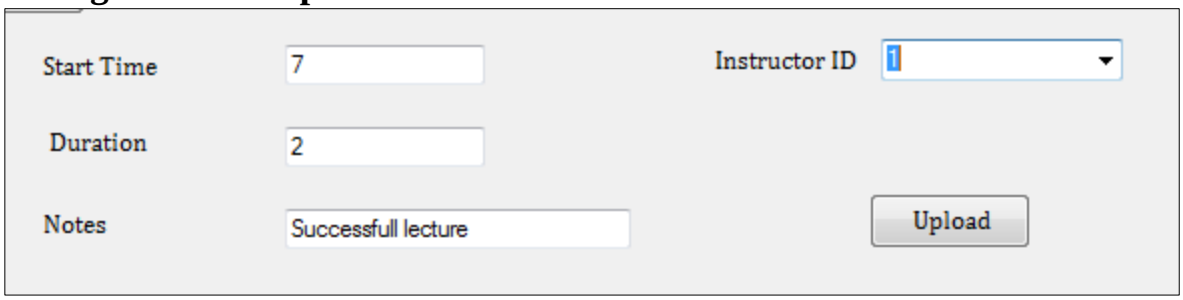


Log_ID	Booking_ID	Start_Time	Duration	Subject	Instructor_ID	Note
1	0	PT0S	0	0	2	0
2	0	PT0S	0	0	3	0
4	1	PT8H	3	Chemistry	0	0
5	2	PT11H	2	Chemistry	0	0
6	3	PT18H	1	Chemistry	0	0
7	7	PT12H	4	Western Music	0	0

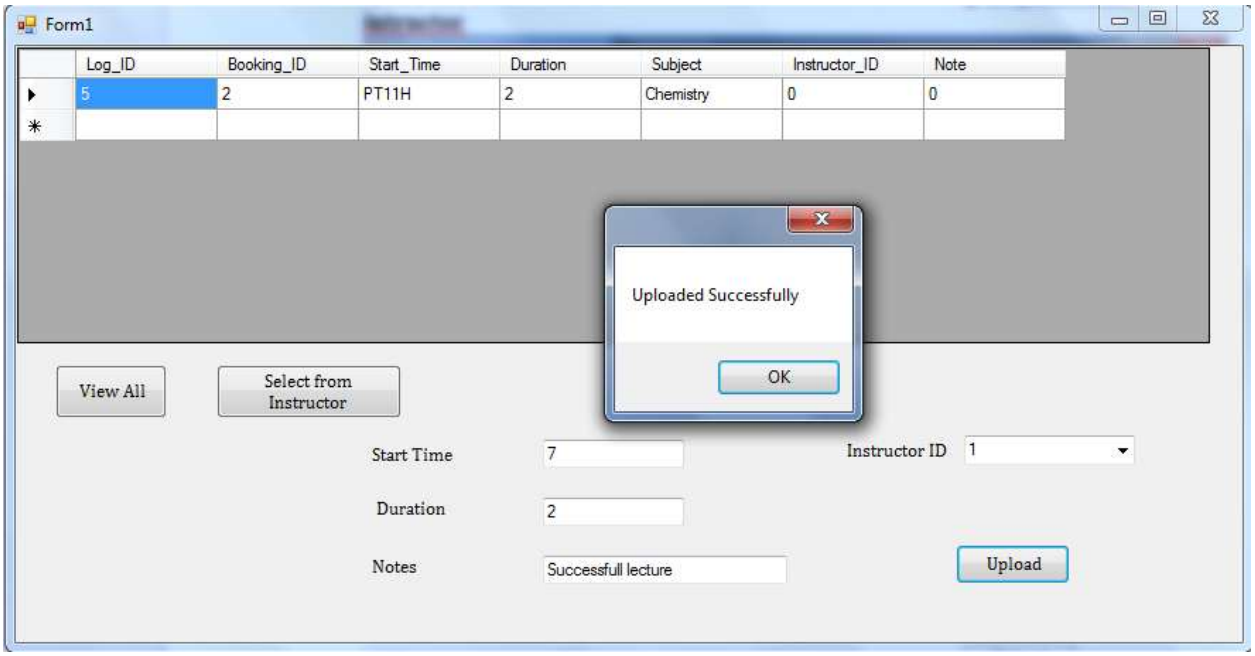
“Select from Instructor” button-returns lessons according to particular Instructor



Saving details of a particular lesson

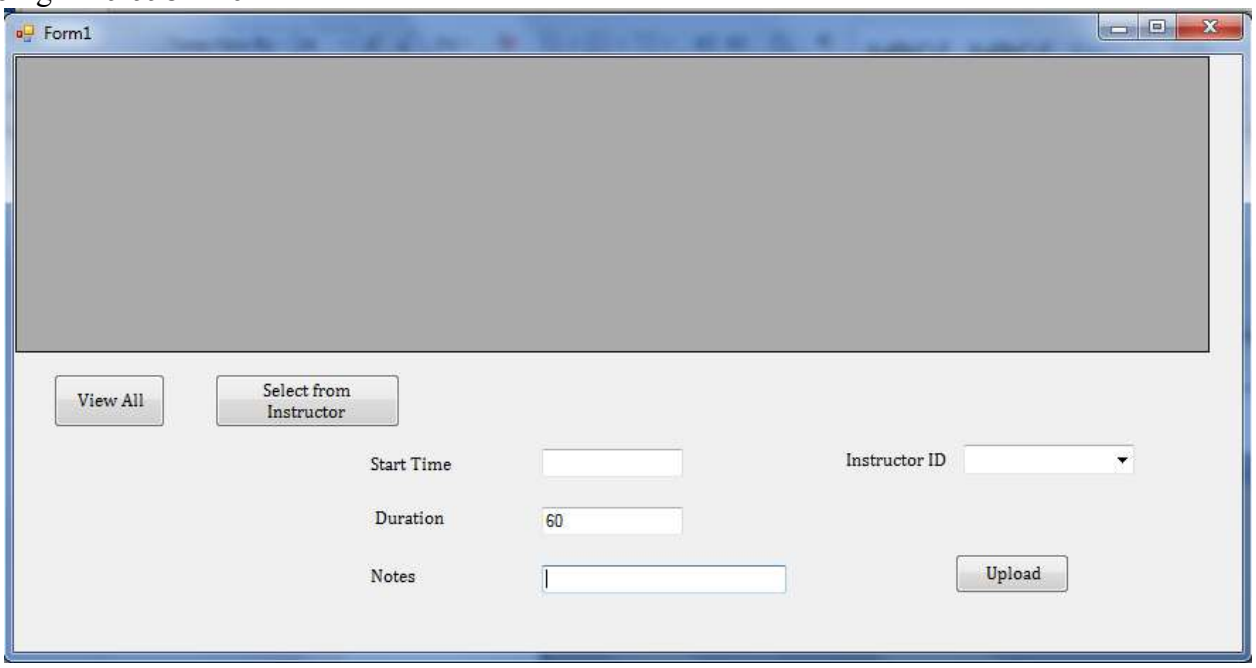


Message when Data is Uploaded Successfully to the lesson manager

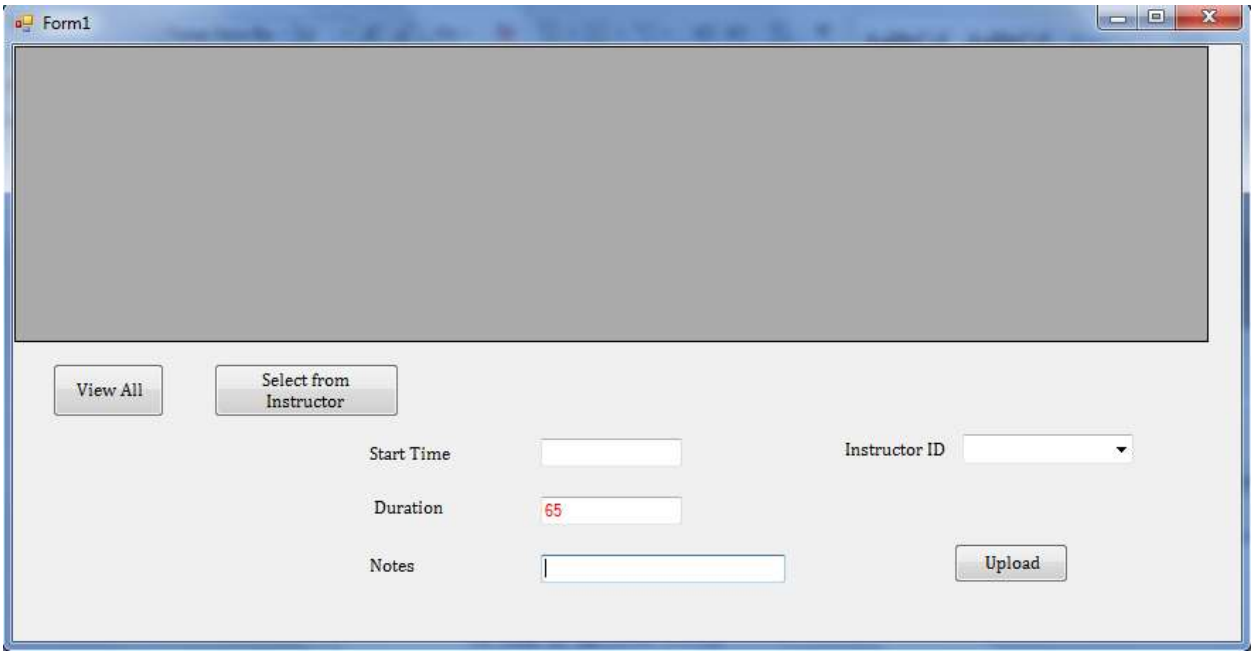


5) Extended Textbox Component

Adding " Duration < 61 "



Adding “ Duration > 61 “



Code of CustomControl1.cs

```
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 ClassLibrary1
{
    public partial class CustomControl1 : TextBox
    {
        private int limit;

        public int Limit
        {
            get { return limit; }
            set { limit = value; }
        }
        public CustomControl1()
        {
            InitializeComponent();
            this.TextChanged += new EventHandler(textBox1_TextChanged);
        }

        protected override void OnPaint(PaintEventArgs pe)
        {
            base.OnPaint(pe);
        }
```

```
        private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
        {
            //MessageBox.Show("Hi");
        }

        private void textBox1_TextChanged(object sender, EventArgs e)
        {
            int val = Convert.ToInt32(this.Text);
            if (val > limit)
                this.ForeColor = Color.Red;
            else
                this.ForeColor = Color.Black;
        }
    }
}
```

6) RESTful Web Service

C# Code of Service1.svc.cs

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

namespace WcfService2
{
    // NOTE: You can use the "Rename" command on the "Refactor" menu to change the class
name "Service1" in code, svc and config file together.
    // NOTE: In order to launch WCF Test Client for testing this service, please select
Service1.svc or Service1.svc.cs at the Solution Explorer and start debugging.
    public class Service1 : IService1
    {
        dbconn conObj = new dbconn();

        public string GetData(int value)
        {
            return string.Format("You entered: {0}", value);
        }

        public List<Booking> allBooking()
        {
            string query = "SELECT * FROM booking";
            DataSet ds = conObj.execQuery(query);
            DataTable dt = new DataTable();
            dt = ds.Tables[0];
            List<Booking> result = new List<Booking>();

            foreach (DataRow dr in dt.Rows)
            {
                Booking bk = new Booking
                {
                    Booking_ID = dr["Booking_ID"].ToString(),
                    Subjects = dr["Subjects"].ToString(),
                    Booking_Date = dr["Booking_Date"].ToString(),
                    Booked_Date = dr["Booked_Date"].ToString(),
                    Maximum_Hourly_Rate = dr["Maximum_Hourly_Rate"].ToString(),
                    Start_Time = dr["Start_Time"].ToString(),
                    Duration = dr["Duration"].ToString(),
                    CustomerID = dr["CustomerID"].ToString(),
                    Day = dr["Day"].ToString(),

                };
                result.Add(bk);
            }
            return result;
        }
```

```
        public List<Booking> allBookingCustomer(string customerID)
        {
            string query = "SELECT * FROM booking where CustomerID = '" + customerID +
"'";
            DataSet ds = conObj.execQuery(query);
            DataTable dt = new DataTable();
            dt = ds.Tables[0];
            List<Booking> result = new List<Booking>();

            foreach (DataRow dr in dt.Rows)
            {
                Booking bk = new Booking
                {
                    Booking_ID = dr["Booking_ID"].ToString(),
                    Booking_Date = dr["Booking_Date"].ToString(),
                    Maximum_Hourly_Rate = dr["Maximum_Hourly_Rate"].ToString(),
                    Duration = dr["Duration"].ToString(),
                    Subjects = dr["Subjects"].ToString(),
                    CustomerID = dr["CustomerID"].ToString(),
                    Start_Time = dr["Start_Time"].ToString(),
                    Day = dr["Day"].ToString()
                };
                result.Add(bk);
            }
            return result;
        }

        public List<user> allCus()
        {
            string query = "SELECT * FROM user";
            DataSet ds = conObj.execQuery(query);
            DataTable dt = new DataTable();
            dt = ds.Tables[0];
            List<user> result = new List<user>();

            foreach (DataRow dr in dt.Rows)
            {
                user us = new user
                {
                    UserID = dr["User_ID"].ToString(),
                    UserName = dr["User_Name"].ToString(),
                    UserType = dr["User_Type"].ToString(),
                    First_name = dr["First_Name"].ToString(),
                    Last_name = dr["Last_Name"].ToString(),
                    Address = dr["Address"].ToString(),
                    Gender = dr["Gender"].ToString(),
                    Dob=dr["Date_Of_Birth"].ToString(),
                    Tp = dr["TP_No"].ToString(),
                  };
                result.Add(us);
            }
            return result;
        }

        public List<user> allCusType(string userType)
        {
            string query = "SELECT * FROM user where User_Type = '" + userType + "'";
            DataSet ds = conObj.execQuery(query);
```

```
            DataTable dt = new DataTable();
            dt = ds.Tables[0];
            List<user> result = new List<user>();

            foreach (DataRow dr in dt.Rows)
            {
                user us = new user
                {
                    UserID = dr["User_ID"].ToString(),
                    UserName = dr["User_Name"].ToString(),
                    UserType = dr["User_Type"].ToString(),
                    First_name = dr["First_Name"].ToString(),
                    Last_name = dr["Last_Name"].ToString(),
                    Address = dr["Address"].ToString(),
                    Gender = dr["Gender"].ToString(),
                    Dob = dr["Date_Of_Birth"].ToString(),
                    Tp = dr["TP_No"].ToString(),
                };
                result.Add(us);
            }
            return result;
        }

        public List<user> allCusTypeEmail(string userType, string email)
        {
            string query = "SELECT * FROM user where User_Type = '" + userType + "'and
User_Name= '" + email + "'";
            DataSet ds = conObj.execQuery(query);
            DataTable dt = new DataTable();
            dt = ds.Tables[0];
            List<user> result = new List<user>();

            foreach (DataRow dr in dt.Rows)
            {
                user us = new user
                {
                    UserID = dr["User_ID"].ToString(),
                    UserName = dr["User_Name"].ToString(),
                    UserType = dr["User_Type"].ToString(),
                };
                result.Add(us);
            }
            return result;
        }

        public List<Invoice> allInvoice()
        {
            string query = "SELECT * FROM invoice";
            DataSet ds = conObj.execQuery(query);
            DataTable dt = new DataTable();
            dt = ds.Tables[0];
            List<Invoice> result = new List<Invoice>();

            foreach (DataRow dr in dt.Rows)
            {
                Invoice inv = new Invoice
                {
                    Invoice_ID = dr["Invoice_ID"].ToString(),
```

```
                    Customer_ID = dr["Customer_ID"].ToString(),
                    Month = dr["Month"].ToString(),
                    Payment_Value = dr["Payment_Value"].ToString(),
                    Payment_Status = dr["Payment_Status"].ToString(),
                 };
                result.Add(inv);
            }
            return result;
        }

        public List<Invoice> allInvoiceCus(string customerID)
        {
            string query = "SELECT * FROM invoice where Customer_ID = '" + customerID +
"'";
            DataSet ds = conObj.execQuery(query);
            DataTable dt = new DataTable();
            dt = ds.Tables[0];
            List<Invoice> result = new List<Invoice>();

            foreach (DataRow dr in dt.Rows)
            {
                Invoice inv = new Invoice
                {
                    Invoice_ID = dr["Invoice_ID"].ToString(),
                    Customer_ID = dr["Customer_ID"].ToString(),
                    Month = dr["Month"].ToString(),
                    Payment_Value = dr["Payment_Value"].ToString(),
                    Payment_Status = dr["Payment_Status"].ToString(),
                };
                result.Add(inv);
            }
            return result;
        }

        public List<Invoice> allInvoiceCusMonth(string customerID, string month)
        {
            string query = "SELECT * FROM invoice where Customer_ID = '" + customerID +
"'and Month= '" + month + "'";
            DataSet ds = conObj.execQuery(query);
            DataTable dt = new DataTable();
            dt = ds.Tables[0];
            List<Invoice> result = new List<Invoice>();

            foreach (DataRow dr in dt.Rows)
            {
                Invoice inv = new Invoice
                {
                    Invoice_ID = dr["Invoice_ID"].ToString(),
                    Customer_ID = dr["Customer_ID"].ToString(),
                    Month = dr["Month"].ToString(),
                    Payment_Value = dr["Payment_Value"].ToString(),
                    Payment_Status = dr["Payment_Status"].ToString(),
                };
                result.Add(inv);
            }
            return result;
        }
    }}
```

C# Code of IService1.cs

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

namespace WcfService2
{
    // 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 GetData(int value);

        [OperationContract]
        [WebGet(UriTemplate = "/booking")]
        List<Booking> allBooking();

        [OperationContract]
        [WebGet(UriTemplate = "/booking/{customerID}")]
        List<Booking> allBookingCustomer(string customerID);

        [OperationContract]
        [WebGet(UriTemplate = "/user")]
        List<user> allCus();

        [OperationContract]
        [WebGet(UriTemplate = "/user/{userType}")]
        List<user> allCusType(string userType);

        [OperationContract]
        [WebGet(UriTemplate = "/user/{userType}/{email}")]
        List<user> allCusTypeEmail(string userType, string email);

        [OperationContract]
        [WebGet(UriTemplate = "/invoice")]
        List<Invoice> allInvoice();

        [OperationContract]
        [WebGet(UriTemplate = "/invoice/{customerID}")]
        List<Invoice> allInvoiceCus(string customerID);

        [OperationContract]
        [WebGet(UriTemplate = "/invoice/{customerID}/{month}")]
        List<Invoice> allInvoiceCusMonth(string customerID, string month);
        // TODO: Add your service operations here
    }


    // Use a data contract as illustrated in the sample below to add composite types to
service operations.
    [DataContract]
```

```
public class CompositeType
{
    bool boolValue = true;
    string stringValue = "Hello ";

    [DataMember]
    public bool BoolValue
    {
        get { return boolValue; }
        set { boolValue = value; }
    }

    [DataMember]
    public string StringValue
    {
        get { return stringValue; }
        set { stringValue = value; }
    }
}

[DataContract]
public class customer
{
    string customerID = "C001";
    string username = "DefaultName";
    string email = "DefaultEmail";


    [DataMember]
    public string CustomerID
    {
        get { return customerID; }
        set { customerID = value; }
    }

    [DataMember]
    public string UserName
    {
        get { return username; }
        set { username = value; }
    }

    [DataMember]
    public string Email
    {
        get { return email; }
        set { email = value; }
    }
}

[DataContract]
public class user
{
    string userID = "001";
    string userName = "DefaultName";
    string userType = "DefaultType";
    string first_name = "DefaultFname";
    string last_name = "DefaultLname";
```

```
string address = "DefaultAdd";
string gender = "DefaultGender";
string dob = "Defaultdob";
string tp = "DefaultTP";

[DataMember]
public string UserID
{
    get { return userID; }
    set { userID = value; }
}

[DataMember]
public string UserName
{
    get { return userName; }
    set { userName = value; }
}

[DataMember]
public string UserType
{
    get { return userType; }
    set { userType = value; }
}

[DataMember]
public string First_name
{
    get { return first_name; }
    set { first_name = value; }
}

[DataMember]
public string Last_name
{
    get { return last_name; }
    set { last_name = value; }
}

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

[DataMember]
public string Gender
{
    get { return gender; }
    set { gender = value; }
}

[DataMember]
public string Tp
{
    get { return tp; }
```

```
        set { tp = value; }
    }

    [DataMember]
    public string Dob
    {
        get { return dob; }
        set { dob = value; }
    }
}

[DataContract]
public class instructor
{
    string instructorID = "I001";
    string hourly_rate = "DefaultRate";

    [DataMember]
    public string InstructorID
    {
        get { return instructorID; }
        set { instructorID = value; }
    }

    [DataMember]
    public string Hourly_rate
    {
        get { return hourly_rate; }
        set { hourly_rate = value; }
    }

}


[DataContract]
public class Booking
{
    string booking_ID = "C001";
    string subjects = "Sinhala";
    string year = "2016";
    string month = "sep";
    string date = "22";
    string booking_Date = "DefaultDate";
    string maximum_Hourly_Rate = "DefaultMonth";
    string duration = "DefaultMonth";
    string customerID = "C001";
    string day = "";
    string start_Time = "";
    string instructor_ID="";


    public string Day
    {
        get { return day; }
        set { day = value; }
    }
```

```
public string Start_Time
{
    get { return start_Time; }
    set { start_Time = value; }
}


[DataMember]
public string Booking_ID
{
    get { return booking_ID; }
    set { booking_ID = value; }
}

[DataMember]
public string Instructor_ID
{
    get { return instructor_ID; }
    set { instructor_ID = value; }
}

[DataMember]
public string CustomerID
{
    get { return customerID; }
    set { customerID = value; }
}

[DataMember]
public string Booking_Date
{
    get { return booking_Date; }
    set { booking_Date = value; }
}

[DataMember]
public string Duration
{
    get { return duration; }
    set { duration = value; }
}

[DataMember]
public string Booked_Date
{
    get { return booking_Date; }
    set { booking_Date = value; }
}

[DataMember]
public string Subjects
{
    get { return subjects; }
    set { subjects = value; }
}

[DataMember]
public string Maximum_Hourly_Rate
```

```
        {
            get { return maximum_Hourly_Rate; }
            set { maximum_Hourly_Rate = value; }
        }
    }

    [DataContract]
    public class Invoice
    {
        string invoice_ID = "";
        public string Invoice_ID
        {
            get { return invoice_ID; }
            set { invoice_ID = value; }
        }

        string customer_ID = "";
        public string Customer_ID
        {
            get { return customer_ID; }
            set { customer_ID = value; }
        }

        string month = "sep";
        public string Month
        {
            get { return month; }
            set { month = value; }
        }

        string payment_Value = "";
        public string Payment_Value
        {
            get { return payment_Value; }
            set { payment_Value = value; }
        }

        string payment_Status = "";
        public string Payment_Status
        {
            get { return payment_Status; }
            set { payment_Status = value; }
        }

    }

}
```

Code of dbconn.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using MySql.Data.MySqlClient;
using MySql.Data;
using System.Data;

namespace WcfService2
{
    public class dbconn
    {
        string connectionString = null;
        public MySqlConnection con;

        public dbconn()
        {
            connectionString = "Database=sowpassignmentnew; Data Source=localhost; User
Id=root; Password=123";
            try
            {
                con = new MySqlConnection(connectionString);
                con.Open();
                Console.WriteLine("Connection Opened");

            }
            catch (Exception ex)
            {
                Console.WriteLine("Connection Open failed");
            }
        }

        public DataSet execQuery(string query)
        {
            DataSet results = new DataSet();
            try
            {
                MySqlDataAdapter sql = new MySqlDataAdapter(query, connectionString);
                sql.Fill(results);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Error : " + ex);
                System.Diagnostics.Debug.Write("Error : " + ex);
            }
            return results;
        }


        public int execNonQuery(string query)
        {
            int condition = -1;
            try
            {
                MySqlCommand cmd1 = new MySqlCommand(query, con);
                condition = cmd1.ExecuteNonQuery();
```

```
            }
            catch (Exception ex)
            {
                Console.WriteLine("Error :" + ex);
                System.Diagnostics.Debug.Write("Error :" + ex);
            }
            finally
            {
                con.Close();
            }

            return condition;
        }
    }
}
```

Code of Service1.svc

```
<%@ ServiceHost Language="C#" Debug="true" Service="WcfService2.Service1"
CodeBehind="Service1.svc.cs"
Factory="System.ServiceModel.Activation.WebServiceHostFactory" %>
```

Results of Methods

localhost:3720/Service1.svc/booking

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
▼<ArrayOfBooking xmlns="http://schemas.datacontract.org/2004/07/WcfService2" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  ▼<Booking>
     <Booked_Date>9/27/2016 8:17:21 AM</Booked_Date>
     <Booking_Date>9/27/2016 8:17:21 AM</Booking_Date>
     <Booking_ID>1</Booking_ID>
     <CustomerID>1</CustomerID>
     <Duration>3</Duration>
     <Instructor_ID/>
     <Maximum_Hourly_Rate>788</Maximum_Hourly_Rate>
     <Subjects>Chemistry</Subjects>
   </Booking>
  ▼<Booking>
     <Booked_Date>10/18/2016 12:00:00 AM</Booked_Date>
     <Booking_Date>10/18/2016 12:00:00 AM</Booking_Date>
     <Booking_ID>2</Booking_ID>
     <CustomerID>2</CustomerID>
     <Duration>2</Duration>
     <Instructor_ID/>
     <Maximum_Hourly_Rate>150</Maximum_Hourly_Rate>
     <Subjects>Chemistry</Subjects>
   </Booking>
  ▼<Booking>
     <Booked_Date>10/22/2016 12:00:00 AM</Booked_Date>
     <Booking_Date>10/22/2016 12:00:00 AM</Booking_Date>
     <Booking_ID>3</Booking_ID>
     <CustomerID>1</CustomerID>
     <Duration>1</Duration>
     <Instructor_ID/>
     <Maximum_Hourly_Rate>200</Maximum_Hourly_Rate>
     <Subjects>Chemistry</Subjects>
   </Booking>
  ▼<Booking>
     <Booked_Date>10/27/2016 12:00:00 AM</Booked_Date>
     <Booking_Date>10/27/2016 12:00:00 AM</Booking_Date>
     <Booking_ID>7</Booking_ID>
     <CustomerID>1</CustomerID>
     <Duration>4</Duration>
     <Instructor_ID/>
     <Maximum_Hourly_Rate>300</Maximum_Hourly_Rate>
     <Subjects>Western Music</Subjects>
   </Booking>
```

localhost:3720/Service1.svc/booking/1

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
▼<ArrayOfBooking xmlns="http://schemas.datacontract.org/2004/07/WcfService2" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  ▼<Booking>
     <Booked_Date>10/7/2016 10:00:00 AM</Booked_Date>
     <Booking_Date>10/7/2016 10:00:00 AM</Booking_Date>
     <Booking_ID>1</Booking_ID>
     <CustomerID>1</CustomerID>
     <Duration>3</Duration>
     <Instructor_ID/>
     <Maximum_Hourly_Rate>788</Maximum_Hourly_Rate>
     <Subjects>Chemistry</Subjects>
   </Booking>
  ▼<Booking>
     <Booked_Date>10/9/2016 9:47:00 AM</Booked_Date>
     <Booking_Date>10/9/2016 9:47:00 AM</Booking_Date>
     <Booking_ID>3</Booking_ID>
     <CustomerID>1</CustomerID>
     <Duration>1</Duration>
     <Instructor_ID/>
     <Maximum_Hourly_Rate>200</Maximum_Hourly_Rate>
     <Subjects>Chemistry</Subjects>
   </Booking>
  ▼<Booking>
     <Booked_Date>10/17/2016 7:19:00 AM</Booked_Date>
     <Booking_Date>10/17/2016 7:19:00 AM</Booking_Date>
     <Booking_ID>7</Booking_ID>
     <CustomerID>1</CustomerID>
     <Duration>4</Duration>
     <Instructor_ID/>
     <Maximum_Hourly_Rate>300</Maximum_Hourly_Rate>
     <Subjects>Western Music</Subjects>
   </Booking>
 </ArrayOfBooking>
```


localhost:3720/Service1.svc/user

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
▼<ArrayOfuser xmlns="http://schemas.datacontract.org/2004/07/WcfService2" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  ▼<user>
     <Address>No,57, Madelgamuwa, Gampaha</Address>
     <Dob>2/2/1998 12:00:00 AM</Dob>
     <First_name>Chathurangi</First_name>
     <Gender>Female</Gender>
     <Last_name>Shyalika</Last_name>
     <Tp>0772150269</Tp>
     <UserID>1</UserID>
     <UserName>chathurangijks@gmail.com</UserName>
     <UserType>Instructor</UserType>
   </user>
  ▼<user>
     <Address>No56, Silva Rd, Moratuwa</Address>
     <Dob>2/2/1008 12:00:00 AM</Dob>
     <First_name>Chamani</First_name>
     <Gender>Female</Gender>
     <Last_name>Shiranthika</Last_name>
     <Tp>0111234561</Tp>
     <UserID>2</UserID>
     <UserName>chamanijks2@gmail.com</UserName>
     <UserType>Customer</UserType>
   </user>
  ▼<user>
     <Address>No56,289Road</Address>
     <Dob>2/9/9988 12:00:00 AM</Dob>
     <First_name>Jaya</First_name>
     <Gender>Male</Gender>
     <Last_name>Sampath</Last_name>
     <Tp>0774567891</Tp>
     <UserID>3</UserID>
     <UserName>jayakodydw@gmail.com</UserName>
     <UserType>Customer</UserType>
   </user>
```

localhost:3720/Service1.svc/user/Customer

This XML file does not appear to have any style information associated with it. The document tree is shown below.

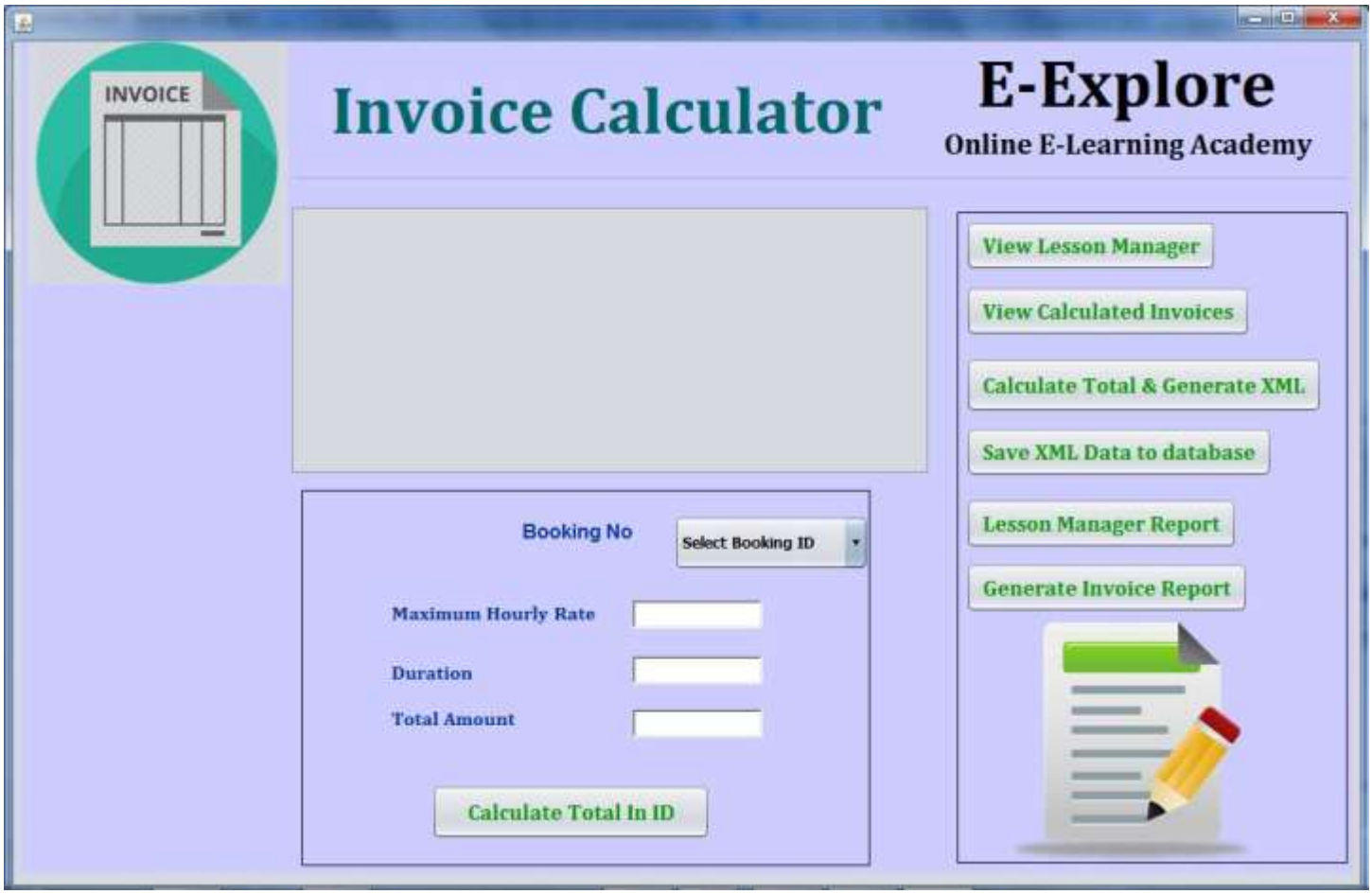

```
▼<ArrayOfuser xmlns="http://schemas.datacontract.org/2004/07/WcfService2" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  ▼<user>
     <Address>No56, Silva Rd, Moratuwa</Address>
     <Dob>2/2/1008 12:00:00 AM</Dob>
     <First_name>Chamani</First_name>
     <Gender>Female</Gender>
     <Last_name>Shiranthika</Last_name>
     <Tp>0111234561</Tp>
     <UserID>2</UserID>
     <UserName>chamanijks2@gmail.com</UserName>
     <UserType>Customer</UserType>
   </user>
  ▼<user>
     <Address>No56,289Road</Address>
     <Dob>2/9/9988 12:00:00 AM</Dob>
     <First_name>Jaya</First_name>
     <Gender>Male</Gender>
     <Last_name>Sampath</Last_name>
     <Tp>0774567891</Tp>
     <UserID>3</UserID>
     <UserName>jayakodydw@gmail.com</UserName>
     <UserType>Customer</UserType>
   </user>
 </ArrayOfuser>
```


localhost:3720/Service1.svc/user/Customer/chamanijks2@gmail.com

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
▼<ArrayOfuser xmlns="http://schemas.datacontract.org/2004/07/WcfService2" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  ▼<user>
     <Address>DefaultAdd</Address>
     <Dob>Defaultdob</Dob>
     <First_name>DefaultFname</First_name>
     <Gender>DefaultGender</Gender>
     <Last_name>DefaultLname</Last_name>
     <Tp>DefaultTP</Tp>
     <UserID>2</UserID>
     <UserName>chamanijks2@gmail.com</UserName>
     <UserType>Customer</UserType>
   </user>
 </ArrayOfuser>
```

7) Invoice Calculator



Main Method-

This GUI is connected with the RESTful web service created in question 6. This code retrieves the booking details as an XML as follows. The result is returned on the console.

```
public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the
default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
```

```
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Invoice.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Invoice.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Invoice.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Invoice.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        }
        //</editor-fold>
    Client client=ClientBuilder.newClient();
        WebTarget
target=client.target("http://localhost:3720/Service1.svc/booking");
        System.out.println(target.request(MediaType.TEXT_XML).get(String.class)


        );

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Invoice().setVisible(true);
            }
        });
```

Output (XML result)

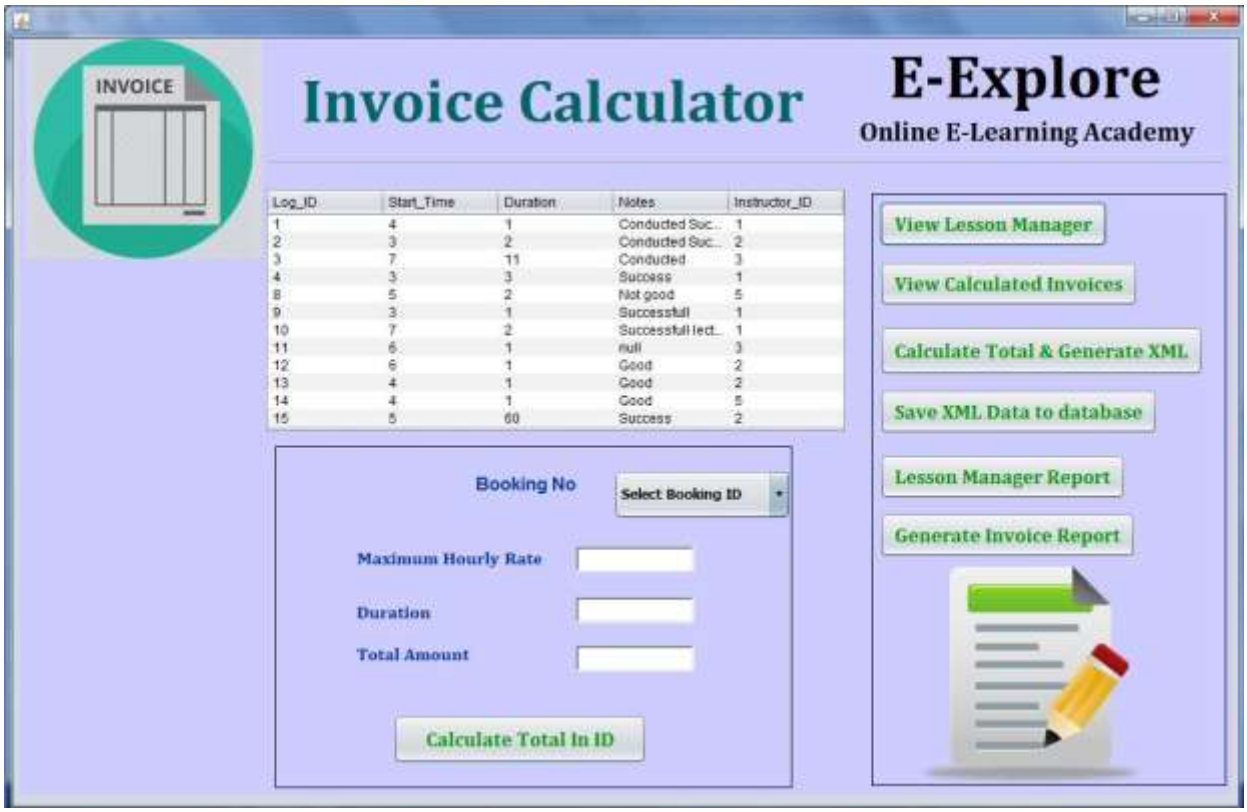

```
run:
<ArrayOfBooking xmlns="http://schemas.datacontract.org/2004/07/WcfService2"
xmlns:i="http://www.w3.org/2001/XMLSchema-
instance"><Booking><Booked_Date>9/27/2016 8:17:21
AM</Booked_Date><Booking_Date>9/27/2016 8:17:21
AM</Booking_Date><Booking_ID>1</Booking_ID><CustomerID>1</CustomerID><Duration>3<
/Duration><Instructor_ID/><Maximum_Hourly_Rate>788</Maximum_Hourly_Rate><Subjects
>Chemistry</Subjects></Booking><Booking><Booked_Date>10/18/2016 12:00:00
AM</Booked_Date><Booking_Date>10/18/2016 12:00:00
AM</Booking_Date><Booking_ID>2</Booking_ID><CustomerID>2</CustomerID><Duration>2<
/Duration><Instructor_ID/><Maximum_Hourly_Rate>150</Maximum_Hourly_Rate><Subjects
>Chemistry</Subjects></Booking><Booking><Booked_Date>10/22/2016 12:00:00
AM</Booked_Date><Booking_Date>10/22/2016 12:00:00
```

```
AM</Booking_Date><Booking_ID>3</Booking_ID><CustomerID>1</CustomerID><Duration>1<
/Duration><Instructor_ID/><Maximum_Hourly_Rate>200</Maximum_Hourly_Rate><Subjects
>Chemistry</Subjects></Booking><Booking><Booked_Date>10/27/2016 12:00:00
AM</Booked_Date><Booking_Date>10/27/2016 12:00:00
AM</Booking_Date><Booking_ID>7</Booking_ID><CustomerID>1</CustomerID><Duration>4<
/Duration><Instructor_ID/><Maximum_Hourly_Rate>300</Maximum_Hourly_Rate><Subjects
>Western Music</Subjects></Booking></ArrayOfBooking>
BUILD SUCCESSFUL (total time: 25 seconds)
```

I. "View Lesson Manager" button click results-

Code:

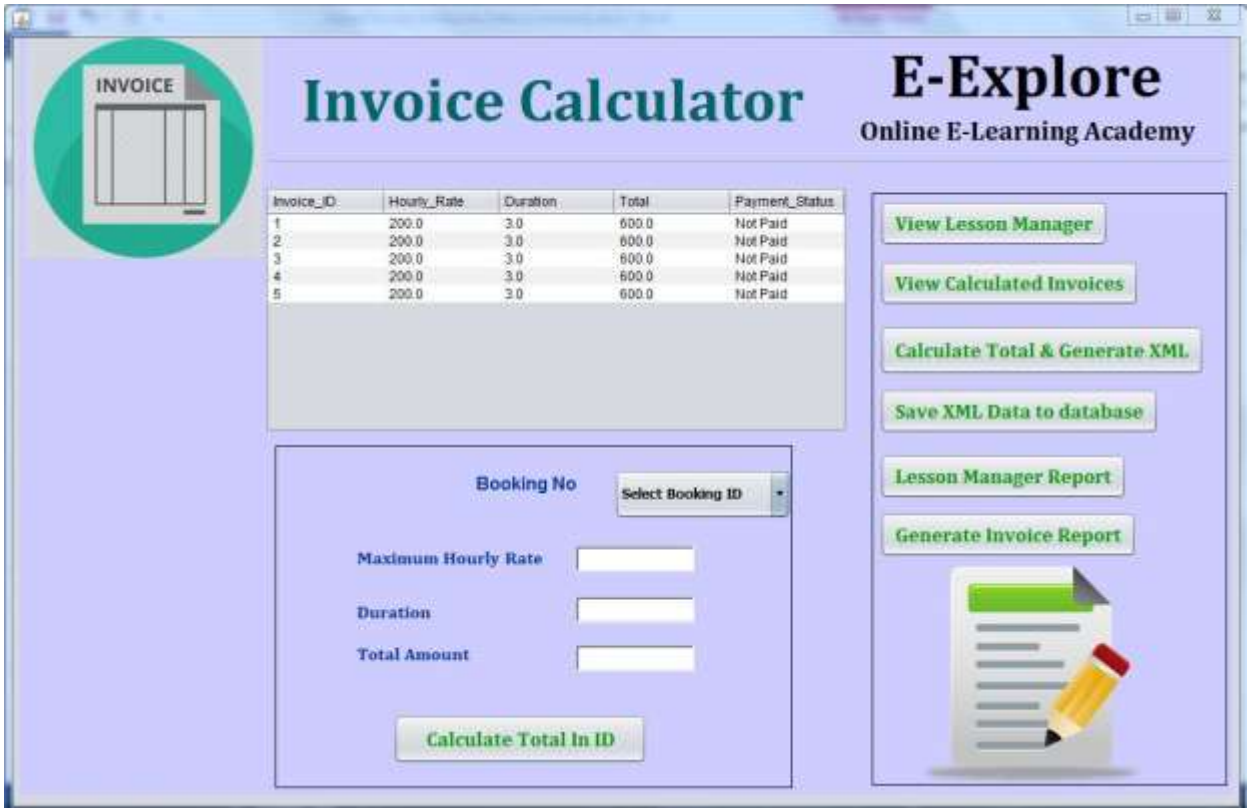
```
public void fetch_data1(){
   try{
      connection1();
      String sql="select * from lessonmanager";
      ResultSet rs=stmt.executeQuery(sql);
      jTable1.setModel(DbUtils.resultSetToTableModel(rs));
   }
   catch(Exception ex){
      JOptionPane.showMessageDialog(null,ex);
   }
}
```



II. "View Calculated Invoices" button click results-

Code:

```
public void fetch_data2(){
   try{
      connection1();
      String sql="select * from invoice";
      ResultSet rs=stmt.executeQuery(sql);
      jTable1.setModel(DbUtils.resultSetToTableModel(rs));
   }
   catch(Exception ex){
      JOptionPane.showMessageDialog(null,ex);
   }
}
```



III. “Calculate Total & Generate XML” button click results-

The Invoice Total of each invoice will be calculated and saved as a XML file. In the console the results will be displayed.

Code:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

    try {
        // TODO add your handling code here:
        //double total_amount=Double.parseDouble(textField1.getText());

        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder builder = factory.newDocumentBuilder();
        org.w3c.dom.Document doc =
builder.parse("http://localhost:3720/Service1.svc/booking");
        XPathFactory xPathfactory = XPathFactory.newInstance();
        XPath xpath = xPathfactory.newXPath();
        XPathExpression expr1= xpath.compile("//ArrayOfBooking/Booking/Duration");
        XPathExpression expr2=
xpath.compile("//ArrayOfBooking/Booking/Maximum_Hourly_Rate");
        XPathExpression expr3= xpath.compile("//ArrayOfBooking/Booking");

        NodeList nl = (NodeList) expr1.evaluate(doc, XPathConstants.NODESET);
        NodeList n2 = (NodeList) expr2.evaluate(doc, XPathConstants.NODESET);
        NodeList n3 = (NodeList) expr3.evaluate(doc, XPathConstants.NODESET);

        AllInvoices inv = new AllInvoices();
        AllInvoiceList invlist = new AllInvoiceList();

        int n4=n3.getLength();
        for (int i = 0; i < n4; i++)

        {
            org.w3c.dom.Node node = nl.item(i);
            String key1 = node.getTextContent();
            dur=Double.parseDouble(key1);

            org.w3c.dom.Node node1 = n2.item(i);
            String key2 = node1.getTextContent();
            hr=Double.parseDouble(key2);

            System.out.println("Duration : "+dur);
            inv.setDuration(dur);
            System.out.println("Maximum Hourly Rate :"+hr);
            inv.setHourly_rate(hr);
```

```
            System.out.println("Invoice Total :"+hr*dur);
            inv.setTotal(hr*dur);

            invlist.add(inv);
        }

        try {
            File file = new File("D:\\InvoiceList.xml");
            JAXBContext jaxbContext = JAXBContext.newInstance(AllInvoiceList.class);
            Marshaller jaxbMarshaller = jaxbContext.createMarshaller();
            jaxbMarshaller.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, true);
            jaxbMarshaller.marshal(invlist, file);
            jaxbMarshaller.marshal(invlist, System.out);
        } catch (JAXBException e) {
            e.printStackTrace();
        }

    } catch (ParserConfigurationException ex) {
        Logger.getLogger(Invoice.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SAXException ex) {
        Logger.getLogger(Invoice.class.getName()).log(Level.SEVERE, null, ex);
    } catch (IOException ex) {
        Logger.getLogger(Invoice.class.getName()).log(Level.SEVERE, null, ex);
    } catch (XPathExpressionException ex) {
        Logger.getLogger(Invoice.class.getName()).log(Level.SEVERE, null, ex);
    }

}
```

AllInvoices.java

```
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement
public class AllInvoices {

    double duration = 0;
    double hourly_rate = 0;
    double total=0;

        public double getDuration() {
                return duration;
        }

        @XmlElement
        public void setDuration(double duration) {
```

```
                this.duration = duration;
        }

    public double getHourly_rate() {
                return hourly_rate;
        }

        @XmlElement
        public void setHourly_rate(double hourly_rate) {
                this.hourly_rate = hourly_rate;
        }

    public double getTotal() {
                return total;
        }

        @XmlElement
        public void setTotal(double total) {
                this.total = total;
        }
    }
```

AllInvoiceList.java

```
import java.util.List;
import java.util.ArrayList;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;


@XmlRootElement(name="InvoiceList")
public class AllInvoiceList {
  List<AllInvoices> allInvoiceList;

   /**
    *
    * @param allInvoiceList
    */

   @XmlElement( name = "Invoice" )
   public void setAllInvoiceList(List<AllInvoices> allInvoiceList)
   {
      this.allInvoiceList = allInvoiceList;
   }

   public List<AllInvoices> getAllInvoiceList()
```

```
    {
        return allInvoiceList;
    }

    public void add( AllInvoices allInvoices )
{
if( this.allInvoiceList == null )
{
this.allInvoiceList = new ArrayList<AllInvoices>();
}

this.allInvoiceList.add( allInvoices );
}

}
```

Output (in console):

```
run:

Duration : 3.0
Maximum Hourly Rate :788.0
Invoice Total :2364.0

Duration : 2.0
Maximum Hourly Rate :150.0
Invoice Total :300.0
Duration : 1.0

Maximum Hourly Rate :200.0
Invoice Total :200.0
Duration : 4.0

Maximum Hourly Rate :300.0
Invoice Total :1200.0
Duration : 3.0

Maximum Hourly Rate :200.0
Invoice Total :600.0

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InvoiceList>
    <Invoice>
        <duration>3.0</duration>
        <hourly_rate>200.0</hourly_rate>
        <total>600.0</total>
    </Invoice>
    <Invoice>
        <duration>3.0</duration>
        <hourly_rate>200.0</hourly_rate>
        <total>600.0</total>
    </Invoice>
    <Invoice>
```

```
            <duration>3.0</duration>
            <hourly_rate>200.0</hourly_rate>
            <total>600.0</total>
        </Invoice>
        <Invoice>
            <duration>3.0</duration>
            <hourly_rate>200.0</hourly_rate>
            <total>600.0</total>
        </Invoice>
        <Invoice>
            <duration>3.0</duration>
            <hourly_rate>200.0</hourly_rate>
            <total>600.0</total>
        </Invoice>
    </InvoiceList>
```


IV. "Save XML Data to Database" button click results-

Code:

```
private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
    File fXmlFile = new File("D:\\InvoiceList.xml");
      DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
      DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
      Document doc = dBuilder.parse(fXmlFile);

      doc.getDocumentElement().normalize();
      NodeList nList = doc.getElementsByTagName("Invoice");
    for (int temp = 0; temp < nList.getLength(); temp++) {
            org.w3c.dom.Node nNode = nList.item(temp);
            if (nNode.getNodeType() == org.w3c.dom.Node.ELEMENT_NODE) {
                Element eElement = (Element) nNode;
            String
n1=eElement.getElementsByTagName("hourly_rate").item(0).getTextContent();
            String
n2=eElement.getElementsByTagName("duration").item(0).getTextContent();
            String n3=eElement.getElementsByTagName("total").item(0).getTextContent();

            connection1();
            int r=stmt.executeUpdate("insert into
invoices(Hourly_rate,Duration,Total,Payment_Status) values('"+n1+"','"+n2+"','"+n3+"','"+"Not
Paid"+"')");
            }
    }
    JOptionPane.showMessageDialog(null,"Invoice Data Saved Successfully");
} catch (Exception e) {
      e.printStackTrace();
```

}}

Output:

←T→	Invoice_ID	Hourly_Rate	Duration	Total	Payment_Status
Edit Copy Delete	1	200	3	600	Not Paid
Edit Copy Delete	2	200	3	600	Not Paid
Edit Copy Delete	3	200	3	600	Not Paid
Edit Copy Delete	4	200	3	600	Not Paid
Edit Copy Delete	5	200	3	600	Not Paid

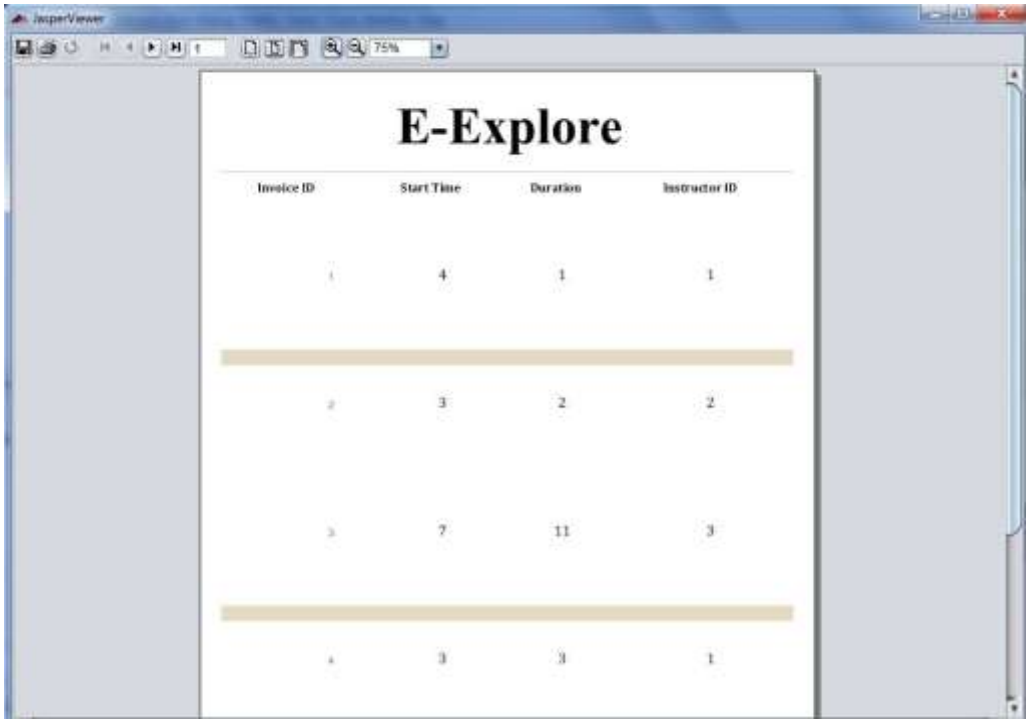
V. "Lesson Manager Report" button click results-

This will display the records of Lesson Manager as a Jasper report.

Code:

```
public void Print(){
   try{
       //Class.forName("com.mysql.jdbc.Driver");
         // Connection
c=DriverManager.getConnection("jdbc:mysql://localhost:3306/sowpassignmentnew",
"root","123");

       String report =
"C:\\Users\\User\\Documents\\NetBeansProjects\\Invoice_Client\\src\\java\\forms\\report3.jrxml
";

       JRXmlDataSource xmlDataSource = new
JRXmlDataSource("D:\\LessonManagerApp.xml","/Users/User");
       JasperReport JASP_REP =JasperCompileManager.compileReport(report);

       JasperPrint JASP_PRINT = JasperFillManager.fillReport(JASP_REP,new
HashMap(),xmlDataSource);
```

```
    JasperViewer.viewReport(JASP_PRINT);
     //JasperExportManager.exportReportToPdfFile(JASP_PRINT, "sample.pdf");

  }
  catch(Exception ex)
  {
    System.out.println(ex);
  }
  }
```



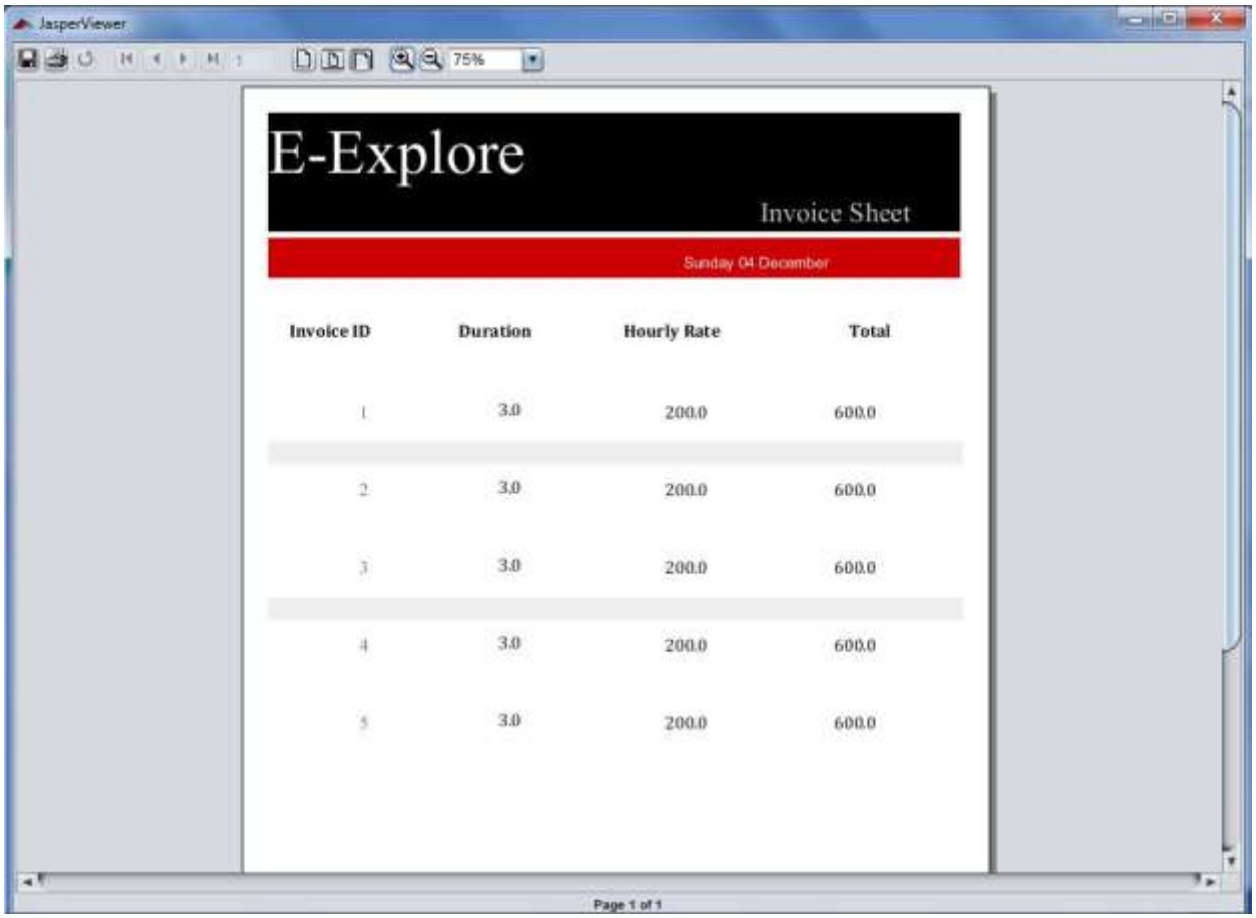
VI. "Generate Invoice Report" button click results-

This will display all the Invoices as a Jasper report.

Code:

```
public void Print_sheet(){
    try{
        //Class.forName("com.mysql.jdbc.Driver");
          // Connection
c=DriverManager.getConnection("jdbc:mysql://localhost:3306/sowpassignmentnew",
"root","123");

        String report =
"C:\\Users\\User\\Documents\\NetBeansProjects\\Invoice_Client\\src\\java\\forms\\report1.jr
xml";
```

```
        JRXmlDataSource xmlDataSource = new
JRXmlDataSource("D:\\InvoiceList.xml","/InvoiceList/Invoice");
        JasperReport JASP_REP =JasperCompileManager.compileReport(report);

        JasperPrint JASP_PRINT = JasperFillManager.fillReport(JASP_REP,new
HashMap(),xmlDataSource);
        JasperViewer.viewReport(JASP_PRINT);
         //JasperExportManager.exportReportToPdfFile(JASP_PRINT, "sample.pdf");
        }
    catch(Exception ex)
    {
        System.out.println(ex);
    }
    }
```



VII. “Calculate Total In ID” button click results-

Code:

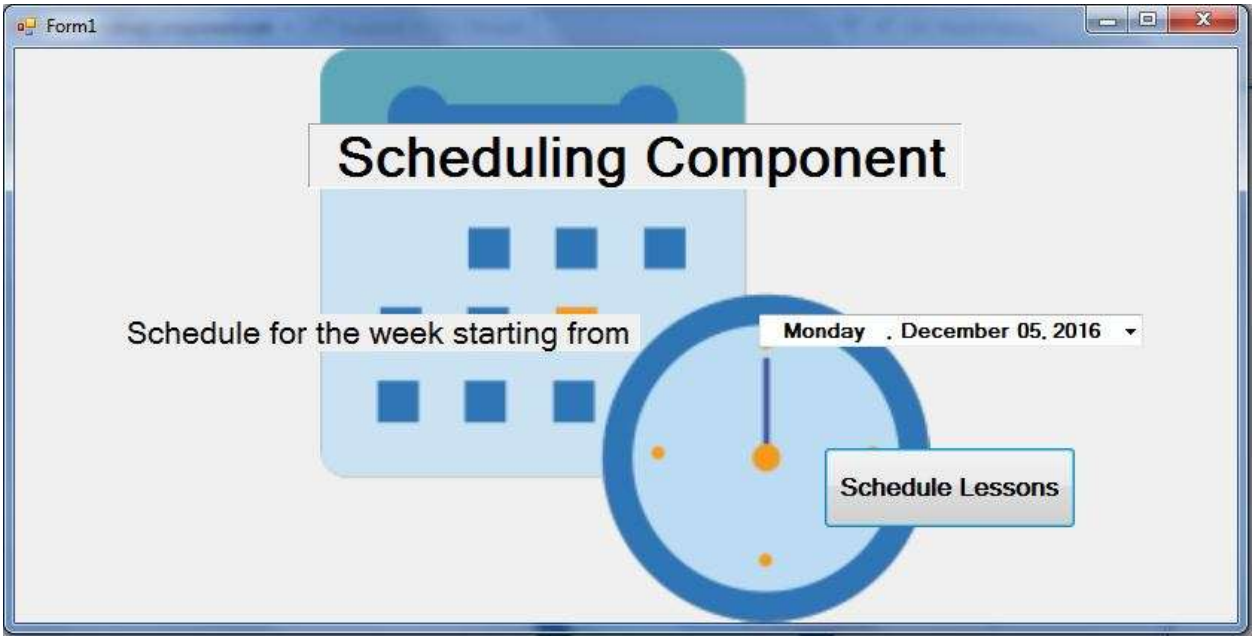
```
private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    double x;
    double y;
    String total;
    try {
        connection1();

        ResultSet r=stmt.executeQuery("select Maximum_Hourly_Rate, Duration from booking
where Booking_ID=" + "'"+jComboBox1.getSelectedItem()+"'");

        if(r.next()){
            textField2.setText(r.getString("Maximum_Hourly_Rate"));
            textField3.setText(r.getString("Duration"));
            x = Double.parseDouble(textField2.getText());
            y = Double.parseDouble(textField3.getText());
            double total1=x*y;
            total = String.valueOf(total1);
            textField4.setText(total);
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
  }
```

When a booking ID is selected and this button is clicked this will retrieve the maximum hourly rate and duration of that invoice and calculate the invoice total.

8) Scheduling Component

I implemented this component in both C#.NET and PHP.

I. C#.NET implementation



Code for the button

```
private void button1_Click(object sender, EventArgs e)
        {
            string connetionString = null;
            MySqlConnection connection;
            MySqlCommand command;
            String sql1 = null;

            MySqlDataAdapter adpter = new MySqlDataAdapter();
            DataSet ds = new DataSet();

            connetionString = "Database=sowpassignmentnew; Data Source=localhost; User
Id=root; Password=123";

            connection = new MySqlConnection(connetionString);
            connection.Open();

           sql1="INSERT INTO lessonmanagerapp(Booking_ID,Start_Time,Duration,Day,Subject)
SELECT Booking_ID,Start_Time,Duration,Day,Subjects FROM booking ; INSERT into
lessonmanagerapp (Instructor_ID) select Instructor_ID from instructor_available_days
where Available_Days IN (select Day from booking where
booking.Day=instructor_available_days.Available_Days)";

            command = new MySqlCommand(sql1, connection);
            adpter.InsertCommand = command;
            adpter.InsertCommand.ExecuteNonQuery();
```

```
MessageBox.Show("Successfuly Scheduled");
connection.Close();
 }
```

When the button is clicked automatically lessons are scheduled according to the given considerations under question 8.

II. PHP implementation

PHP Code for schedule.php

```
<?php
$servername = "localhost";
$username = "root";
$password = "123";
$dbname = "sowpassignmentnew";

$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}

$ScDate = "";
$date1Err = "";

if (isset($_POST['submit'])) {

 if (empty($_POST["ScDate"])) {
   $date1Err = "Date is required";
         }else{
         $date1 = mysqli_real_escape_string($conn, $_POST['ScDate']);
                  }
}

if (!empty($_POST["ScDate"]) && !($date1Err))
{
$sql10="INSERT INTO
lessonmanagerapp(Booking_ID,Start_Time,Duration,Day,Subject)
SELECT Booking_ID,Start_Time,Duration,Day,Subjects FROM booking";

$sql9="INSERT into lessonmanagerapp (Instructor_ID)
select Instructor_ID from instructor_available_days where Available_Days IN (select
Day from booking where booking.Day=instructor_available_days.Available_Days)";

if (mysqli_query($conn, $sql9) && mysqli_query($conn, $sql10)) {
   echo "New Schedule created successfully";
} else {
  echo "Could not able to execute $sql9. " . mysqli_error($conn);
}
}

if (isset($_POST['submitIns'])) {

$sql14="INSERT into lessonmanagerapp (Instructor_ID)
```

```
select Instructor_ID from instructor_available_days where Available_Days IN (select
Day from booking)";


if (mysqli_query($conn, $sql14)) {
    echo "New Instructor scheduled successfully";
} else {
   echo "Could not able to execute $sql14. " . mysqli_error($conn);
}
            }


mysqli_close($conn);

?>
```

Success message after scheduling lessons for a given week