

**GIVE AWAY**  
**FIRST INCREMENT REPORT**

**Submitted By:**

**Sashidhar Reddy Gowra    12428313**

**Yashwant Kumar Palisetty 16202251**

**Ravi Kanth Devanaboyina 16198171**

**Anudeep Reddy Gujjula    16190413**

## Report

### UI Design:

- **User Login Page:** The Login page is the first page of our application where a user needs to give his username and password which he has created for himself in user registration page. This page includes basic html elements like textboxes, buttons, images and certain CSS styles and some scripting functions to validate the user.



**GIVE AWAY!**

*it's*  
giveaway  
*time!*

User Name :

Password :

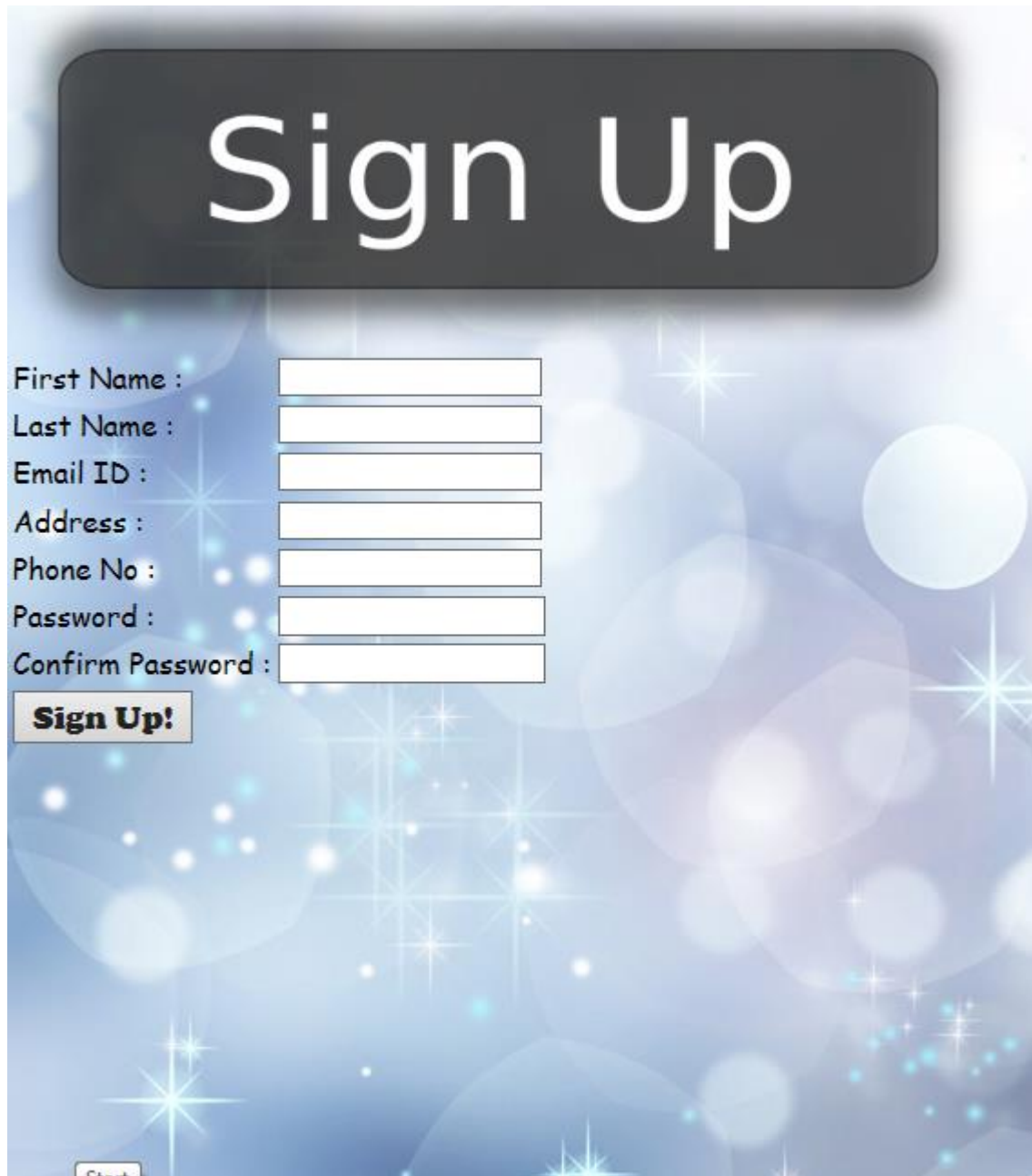
**Log In**

## Not a user??

[Sign Up](#)

- **User Registration Page:** This page is like signup page or user registration page which pops up whenever a user clicks on the signup button on the login page. A user must be a registered user

If he wants to use the application. A user might be a donor or a taker and he needs to fill the following fields like First name, last name, email address, phone number and he can choose a password for himself. On successful registration a success message will be popped on the screen and the registered user can navigate to login page and can give his credentials for logging in.



The image shows a 'Sign Up' form on a blue background with bokeh light effects. At the top, a dark grey rounded rectangle contains the text 'Sign Up' in white. Below this, the form fields are arranged vertically on the left, each followed by a white input box on the right. The fields are: 'First Name :', 'Last Name :', 'Email ID :', 'Address :', 'Phone No :', 'Password :', and 'Confirm Password :'. Below the input boxes is a button labeled 'Sign Up!'. In the bottom left corner, there is a small 'Start' button.

Sign Up

First Name :

Last Name :

Email ID :

Address :

Phone No :

Password :

Confirm Password :

**Sign Up!**

Start

- **Registration Success Page:** This page will have the success message and a button to navigate the user back to login page.

# Thank you for signing up!

Please login to continue!

[Log In](#)

- **Description Page:** This page is like the description page for purpose of the application and intended users who may benefit using this application.

Now a days, students travel to different countries in order to pursue their higher studies, students face a lot of difficulties in getting adjusted to the new environment, cross culture and in various other aspects. The major problem they face is to gather the household items in the initial stages. The basic household items can be arranged by themselves, but when it comes to the point of major ones, it is difficult for them to manage their money on these large household items. In well developed countries like US and UK, there are some non-profit organizations which donate household items to the newly arrived students. But, it is a bit difficult for the students to get in touch with these organizations. So, we need an application, which can bring these organizations and the students closer.

[Back](#)

## DATABASE Design:

- **Creation of Tables and Identification of Constraints:**  
We have created 4 tables namely, Authentication, Registration, GiverItemTable, GrabberItemTable discussed briefly as follows,

**Authentication Table:** This table has userId, userName and password columns and the main purpose for the creation of this table is to store the userName and passwords of the users. UserId

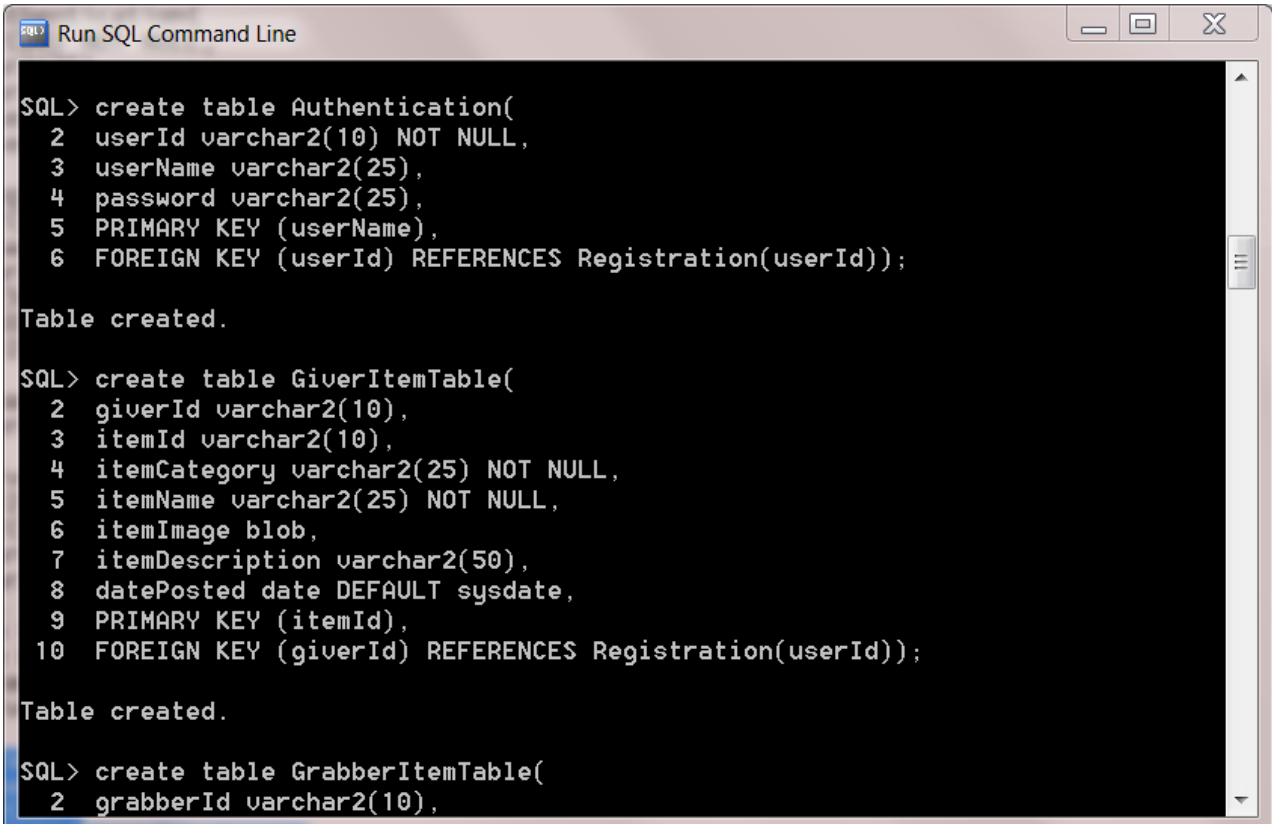
is autogenerated. userName is identified as Primary Key and userId of this table refers to userId of Registration Table.

**Registration Table:** This table has userId, firstName, lastName, email, mobile, address, dob, user\_flag and the main purpose for the creation of this table is to store the details with which the Student/Grabber will be registered in the application. userId in this table is identified as the primary key. User\_flag specifies whether the user is a grabber or a giver.

**GiverItemTable:** This table has giverId, itemId, itemCategory, itemName, itemImage, itemDescription, datePosted. The main purpose of this table is to store the details of the items that the people are willing to donate. Here, itemId is the primary key, and giverId refers to the userId of Registration Table.

**GrabberItemTable:** This table has grabberId, itemId, itemCategory, itemName, itemImage, itemDescription, dateGrabbed. The main purpose of this table is to store the details of the items that the students take. Here, itemId is the primary key and grabberId refers to the userId of the Registration table.

#### Screenshots:



```
SQL> create table Authentication(
  2  userId varchar2(10) NOT NULL,
  3  userName varchar2(25),
  4  password varchar2(25),
  5  PRIMARY KEY (userName),
  6  FOREIGN KEY (userId) REFERENCES Registration(userId));

Table created.

SQL> create table GiverItemTable(
  2  giverId varchar2(10),
  3  itemId varchar2(10),
  4  itemCategory varchar2(25) NOT NULL,
  5  itemName varchar2(25) NOT NULL,
  6  itemImage blob,
  7  itemDescription varchar2(50),
  8  datePosted date DEFAULT sysdate,
  9  PRIMARY KEY (itemId),
  10 FOREIGN KEY (giverId) REFERENCES Registration(userId));

Table created.

SQL> create table GrabberItemTable(
  2  grabberId varchar2(10),
```

```
SQL> Run SQL Command Line
ORA-00906: missing left parenthesis

SQL>
SQL> create table GrabberItemTable(
  2 grabberId varchar2(10),
  3 itemId varchar2(10),
  4 itemCategory varchar2(25) NOT NULL,
  5 itemName varchar2(25) NOT NULL,
  6 itemImage blob,
  7 itemDescription varchar2(50),
  8 dateGrabbed date DEFAULT sysdate,
  9 PRIMARY KEY (itemId),
  10 FOREIGN KEY (grabberId) REFERENCES Registration(userId));

Table created.

SQL>
```

```
SQL> Run SQL Command Line

SQL> create table Registration(
  2 userId varchar2(10),
  3 firstName varchar2(25) NOT NULL,
  4 lastName varchar2(25),
  5 email varchar2(50) NOT NULL,
  6 mobile number(10) NOT NULL,
  7 address varchar2(100) NOT NULL,
  8 dob date,
  9 user_flag varchar2(1),
  10 PRIMARY KEY(userId));

Table created.

SQL> create table Authentication(
  2 userId varchar2(10) NOT NULL,
  3 userName varchar2(25),
  4 password varchar2(25),
  5 PRIMARY KEY (userName),
  6 FOREIGN KEY (userId) REFERENCES Registration(userId));

Table created.

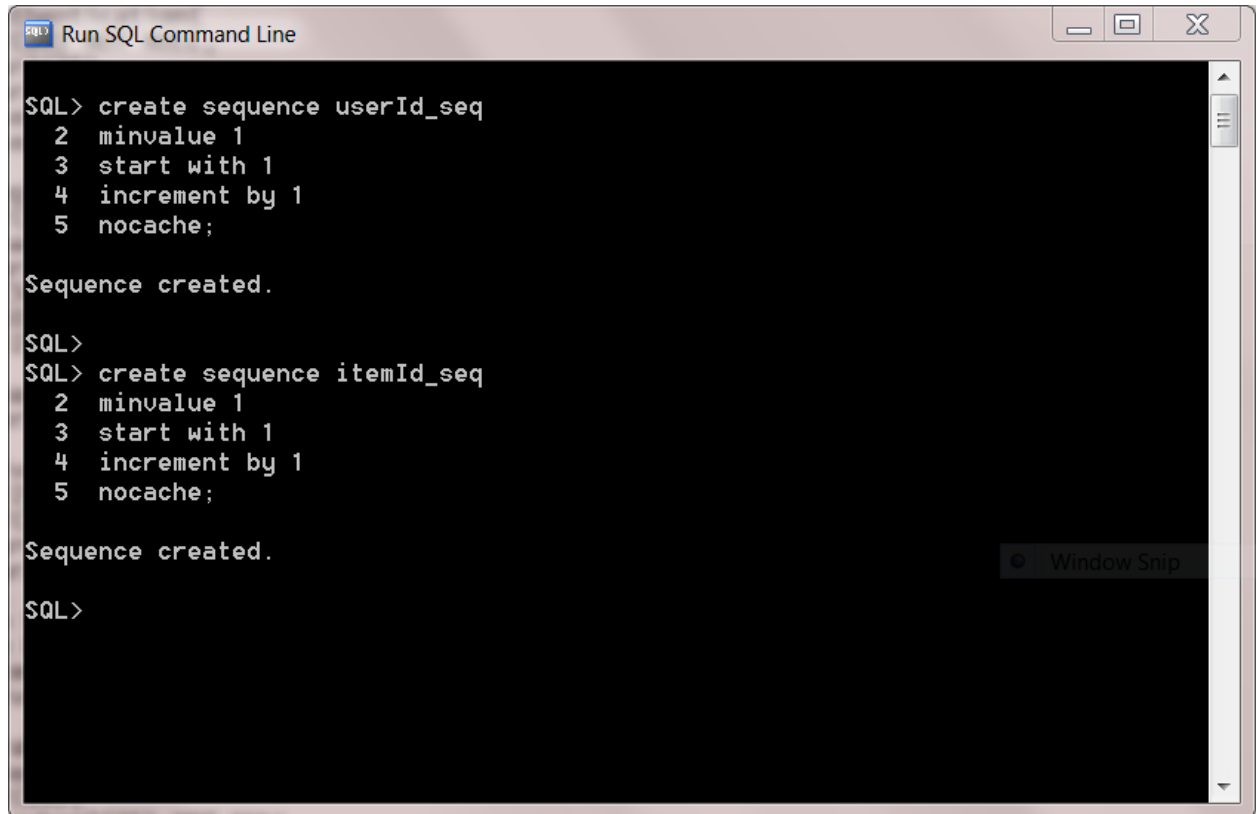
SQL> create table GiverItemTable(
  2 giverId varchar2(10),
```



- **Creating Sequences for Auto Increment:**

For this application we have created two sequences for autogeneration logic for itemId and userId namely, userId\_seq and itemId\_seq.

**Screenshots:**



```
SQL> create sequence userId_seq
2 minvalue 1
3 start with 1
4 increment by 1
5 nocache;

Sequence created.

SQL>
SQL> create sequence itemId_seq
2 minvalue 1
3 start with 1
4 increment by 1
5 nocache;

Sequence created.

SQL>
```

- **Converting Blob files to binary files and Inserting into Database:** When dealing with blob files, they need to be converted into binary files and should be stored in certain location with a filename and then we can insert the generated binary file into the required table. Here we have written a stored procedure for creating a binary file for each Blob file and inserting it into the database.

```

SQL> DECLARE
2  src_lob BFILE := BFILENAME('IMG', 'Koala.jpg');
3  dest_lob BLOB;
4  BEGIN
5  INSERT INTO lob_table VALUES(2, EMPTY_BLOB())
6  RETURNING doc INTO dest_lob;
7
8  DBMS_LOB.OPEN(src_lob, DBMS_LOB.LOB_READONLY);
9  DBMS_LOB.LoadFromFile( DEST_LOB => dest_lob,
10                        SRC_LOB  => src_lob,
11                        AMOUNT    => DBMS_LOB.GETLENGTH(src_lob) );
12  DBMS_LOB.CLOSE(src_lob);
13
14  COMMIT;
15  END;
16  /

PL/SQL procedure successfully completed.

```

- Retrieving the Blob files with Java Code:** Once a binary file is created and inserted into a table we need to access the file using a java code so that we can display the required image on HTML page. So here using Java we established a connection with Oracle Database and using the user credentials we have logged in and retrieved the inserted images and displayed them using MS paint. This is just a start up code for trying successful retrieval of inserted images into Database.

package hackathon;

```

import java.io.ByteArrayOutputStream;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.sql.Blob;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

```

```

public class Hackathon {
public static void main(String args[])
{

```

```

    Connection conn = null;
    String url = "jdbc:oracle:thin:@Shashi-PC:1521:XE";
    String driver = "oracle.jdbc.OracleDriver";
    String userName = "scott";
    String password = "tiger";
    ResultSet rs = null;
    try {
    Class.forName(driver).newInstance();
    conn = DriverManager.getConnection(url,userName,password);

```



```

Statement stmt = conn.createStatement();
rs =stmt.executeQuery("select doc from lob_table");
Blob lob = null;
while (rs.next()) {
    lob=rs.getBlob("doc");
}

InputStream in1 = lob.getBinaryStream();
ByteArrayOutputStream out1 = new ByteArrayOutputStream();
OutputStream outputStream1 = new FileOutputStream("blobImage.png");

int bufferSize = 1024;
int length = (int) lob.length();

byte[] buffer = new byte[bufferSize];

while((length = in1.read(buffer)) != -1)
{
    out1.write(buffer,0,length);
}
out1.writeTo(outputStream1);
in1.close();
conn.close();

Process p1 =Runtime.getRuntime().exec("mspaint blobImage.png");

}catch (Exception e) {
    e.printStackTrace();
}

}
}

```

### Implementation Status Report:

Work completed: UI Design for some of the screens and Database design.

- Description: In the UI part screens like user login page, user registration page, description page, registration success page have designed and with certain CSS styles and required JavaScript. In Database part the design of storage tables with required constraints have been done along with stored procedures to convert a blob image file into binary file and store them into table.
- Responsibility (Task, Person):
  - Sashidhar Reddy Gowra**-developed java code and stored procedure required for inserting and retrieving blob image from Database.
  - Yashwanth Palisetty**: UI Screen Design for Login and Success Pages.
  - Ravikanth Devaboina**: Creating tables and generating auto increment sequences to generate some unique ids.
  - Anudeep Reddy Gujjula**: UI Screen Design for Registration and Description pages.
- Time taken (#hours): UI Design: 12 hrs Database Design: 14 hrs.

- Contributions (members/percentage): Sashidhar Reddy Gowra 25%, Yashwanth Palisetty 25%, Ravikanth Devaboina 25%, Anudeep Reddy Gujjula 25%.

**Work to be completed**

- Description: Populating the UI screens with data from Database, Creating web services to get the data from Database, Using the already existing Google maps API for sharing the address location of a user, Unit Testing, System Testing, Testing the application on Android Devices
- Responsibility (Task, Person):  
Creating Web services, Hash Functions: Sashidhar Reddy Gowra, Ravikanth Devaboina.  
Populating screens, UI alignments: Yashwanth Palisetty, Anudeep Reddy Gujjula.
- Time to be taken (estimated #hours) 150 hrs.
- Issues/Concerns: Creating RESTfull Web services, Creating hash functions for password hiding, Android devices screen alignments.

**Links:**

**GitHub Link:**

<https://github.com/sashi987/ASE/tree/master/Increment1>

**ScrumDo Link:**

<http://www.scrumdo.com/projects/project/giveaway/iteration/121735>