



# UNIVERSITY OF CALGARY

---

## **Course**

Design and Implementation of Database Systems (CPSC 671)

## **Under the guidance of**

*Prof. Reda Alhajj*

# **Implementation of Questionnaire Survey Application**

Submitted by:

Rinku Saru - 30109031

Yadvendar Singh - 30103358

## Table of Content

1. Introduction	4
1.1 Purpose	4
1.2 System Requirements	4
2. Methodology	5
2.1 Front End	5
2.1.1 User-Interface	5
2.1.2 Scripting	8
2.1.3 Unit Testing	8
3. Backend Methodology	9
3.1 Tables	11
3.1.1 users	11
3.1.2 surveys	11
3.1.3 questions	11
3.1.4 locations	11
3.1.5 age_groups	12
3.1.6 survey_agegroups	12
3.1.7 survey locations	12
3.1.8 genders	12
3.2 Endpoints	12
3.2.1 UserRegister	13
3.2.2 UserLogin	13
3.2.3 AgeGroupList	13
3.2.4 LocationList	13
3.2.5 Survey	14
3.2.6 SurveyListByUsername	14
3.2.7 SurveyList	14
3.2.8 QuestionList	15
3.2.9 Retrieve	15
3.2.10 UpdateSurvey	16
3.2.11 SurveyAgeGroupList	16
3.2.12 SurveyLocationList	16
3.2.13 GenderList	17
4. Analysis	18
4.1 Survey Creation	18
4.2 Result	19

## **Abstract**

The aim of the study was to evaluate the usefulness of customer feedback from satisfaction surveys. Surveys containing different questionnaires where users can rate the questions based on the satisfaction. Furthermore, analysis is done from gathering different user information such as age-groups, genders, location to understand the better perspective of the problem.

This report presents an implementation of Questionnaire Survey Application for users to create and analyze survey details. We created Front End in Unity3D Engine using C# programming language , the Back End was developed in Python using Flask, to create REST API. The final build of the app was used to gather data from 25 participants. Gathered data was then analyzed based on age-group, genders, and locations.

# **1.Introduction**

## **1.1 Purpose**

The purpose of this project is to build a Questionnaire Survey Application to provide an interface to the users to create different surveys and analyze the survey details. The scope of the application is divided into two parts: participation and creation. In Participation any user can use the application and participate in any surveys and submit the data to be analyzed.

In Creation the user with admin access can create a survey which later the participant will use and then the creator can use the application to do the analysis of the survey depending on various matrix like questions score, genders participation, different age groups and from different regions.

## **1.2 Motivation**

Feedback is one of the most important aspects of any task. It's a way of communication from one user to another who can use that information to adjust and improve current and future actions and behaviours. Effective feedback and visual analysis could help individuals in taking preventative measures and improving the actions or process.

## **1.3 System Requirements**

The Application consists of two development phases Front-End and Back-End. Front-End of the application is developed in Unity3D version 2020.1.2f1 using C# programming and Backend Restful API is developed using python Flask library and SQLAlchemy for database queries.

The application then can be converted to .apk and .ipa when exporting from Unity for the respective platform by choosing from File/Builds/BuildOptions menu.

Any Android phone with supporting API 18 or above can be used to install the application in respective device and Iphone or Ipad can be used to test the .ipa in the apple device (you need to have the respective apple account and file permission to install the .ipa file in the apple device).

## 2. Methodology

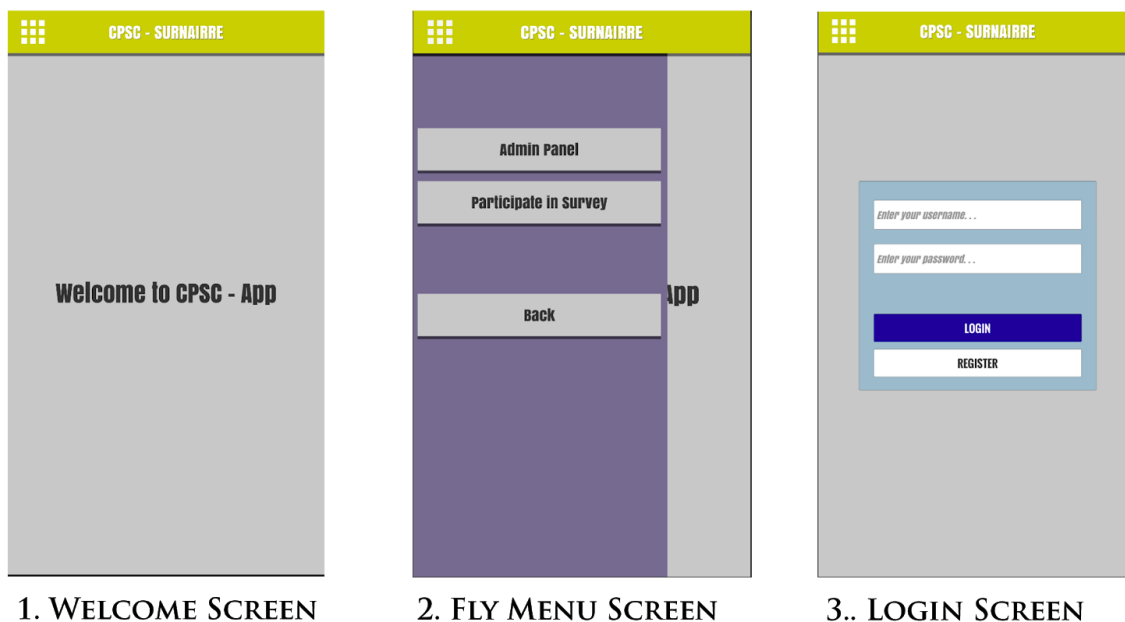
### 2.1 Front-End

Unity3D Engine and C# programming language is used to develop the front-end of the application. Unity is a cross-platform engine developed by Unity Technologies, first announced and released in June 2005 at Apple Inc. Unity is used to develop Games and applications as it provides solutions ready to port to various platforms such as Android, Ios, Window.

#### 2.1.1 User-Interface

Unity Provides a Solutions to use different types of Texture which can be used to build UI using simple drag and drop in Unity Scene window.

These are the various UI panel used in the application :



#### 1. Welcome Screen

This is the Home screen for the application whenever the user launches this app.

#### 2. Fly Menu Screen

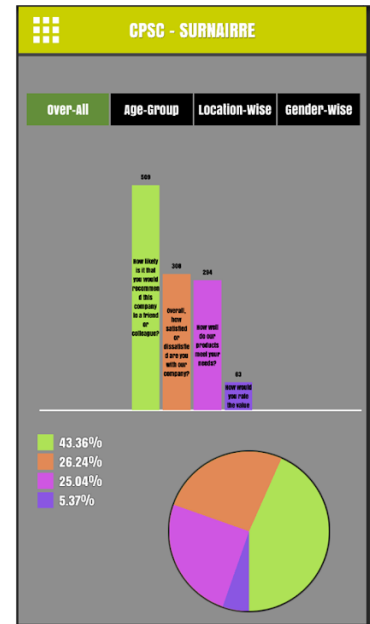
When the user click on the top-left menu button the screen will appear with various button

#### 3. Login Screen

If User clicks on the Admin Panel. he/she needs the proper credential to access the admin panel.

4. REGISTER SCREEN

5. ADMIN SCREEN



6. ANALYSIS SCREEN

#### 4. Register Screen

If a user doesn't have the credential he/she can register to access the admin panel.

#### 5. Admin Screen

After login or Register user is navigated to Admin screen where he/she can analyze the survey if already created or can choose to create the survey

#### 6. Analysis Screen

Users can access this screen on clicking the survey button from Survey Screen which will appear after the user clicks on the analyze button from Admin screen resulting in populating all the survey which was created via only that user.

Users can access various tabs on this screen to view the analysis of the survey based on the participation of various participant users.

CPSC - SURNAIRRE

ENTER SURVEY NAME....

+

CREATE SURVEY

7. CREATE SURVEY SCREEN

CPSC - SURNAIRRE

surveys

customer satisfaction survey

8. SURVEY SCREEN

CPSC - SURNAIRRE

How likely is it that you would recommend this company to a friend or colleague?

Least

1
2
3
4
5-Neutral
6
7
8
9
10

Most

Next

9. QUESTIONNAIRE SCREEN

## 7. Create Survey Screen

Users can access this screen from the create button in Admin Screen. He/she can create a survey adding the survey name and question on click the plus button in the panel.

## 8. Survey Screen

Users can access this screen either by clicking Analyze Button in Admin Screen or Participation in Survey from Fly Menu Screen. This will show all the surveys created so far by any users if the user is participating in the survey otherwise it will show all those surveys created by that user if doing analysis.

## 9. Questionnaire Screen

Questionnaire Screen will be accessed via clicking the any survey button from Survey Screen if the user is participating in the survey. Various questions related to the survey will be asked from the user and the user will rate his/her experience as the answers to those questions.

At the end user can submit the survey report by clicking on the submit button.

### 2.1.2 Scripting

C# programming is used for the scripting in the application and various scripts are created and programmed accordingly to listen to the various actions and events performed by the user and for the communication of the backend server.

For Instance are the UI Panel scripts are under Scripts/UI folder which take care of the all dynamic transition or User interface of the application.

Managerial scripts can be found under the Scripts/Manager folder which mostly uses Singleton pattern and performs all the data communication for various front UI scripts as well as communicating, sending and retrieving data from the server.

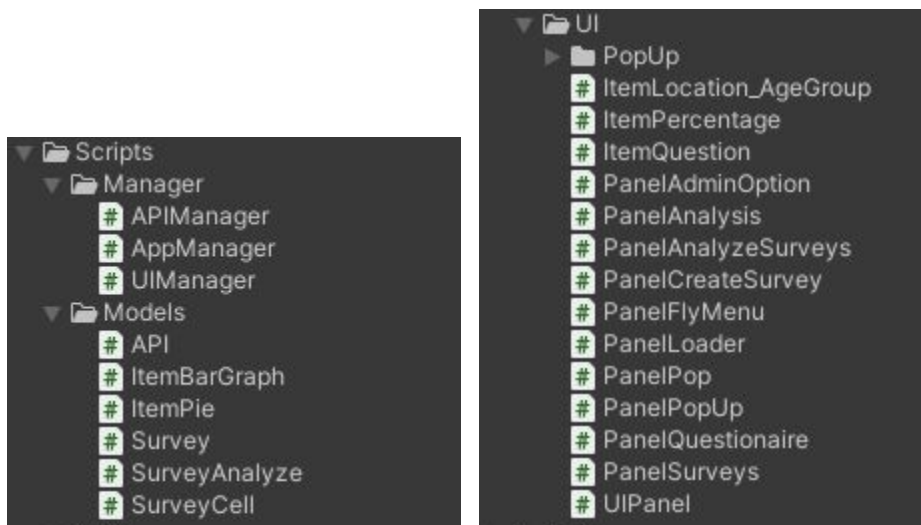


Fig : C# Scripting Architecture

### 2.1.3 Unit Testing

Unity Engine provides a default framework to write the Unit Test script and test it at runtime as well as in Editor Mode. Editor Mode Unit Tests are the tests that can be performed on the static data present in the project and PlayMode Test can be performed to check the dynamic data during the runtime of application.

Because Mostly in the application mainly communication is via the api and it is very important to check the integrity and data that we're sending and receiving by these API. APITest.cs script is created where several Unit Tests are written and later on check if the test fails or passes. All tests must pass which show the data that we're receiving is untouched.



To check that the value are as we expected to be received the unity test framework provides us with several equality assertion for example one of which is:

Assert.AreEqual(obj a, obj b) : Pass if both obj are same otherwise fails.

APITests.cs script can be found under Assets/Test/PlayMode folder and attach are the screenshot of the test written and they pass which is shown by the green tick.

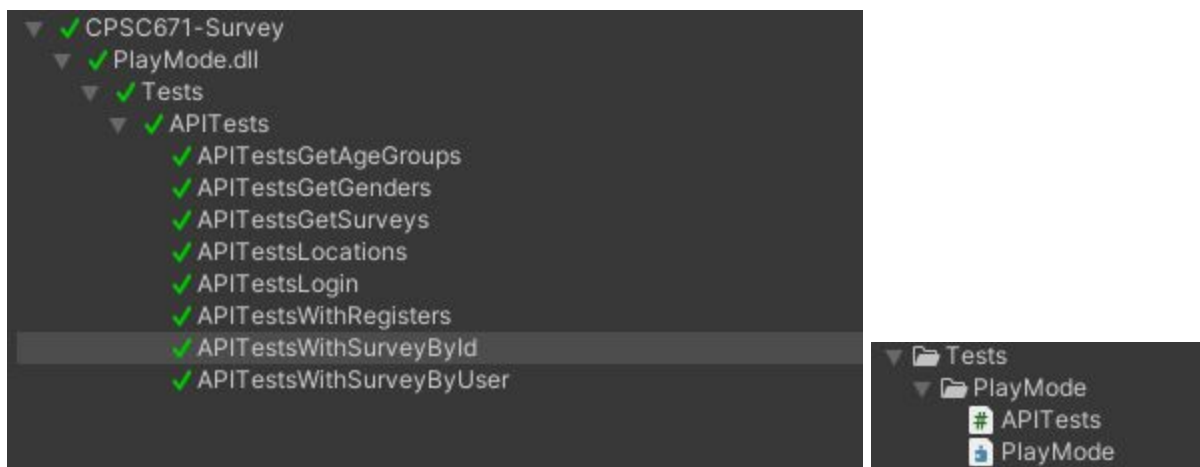


Figure : Test Runner screen

### 3. BACKEND Methodology

The backend has been implemented using Flask and SQLAlchemy in python to generate REST API. Some dependencies that are mentioned in the Readme file need to be installed first for running the API locally. The Database consists of the below mentioned table with their attributes. Entity relation diagram is also attached below the table, which shows the type of relation between tables, and its attributes.

Table Name	Attributes	Primary Key/ Foreign Key
users	<ol style="list-style-type: none"> <li>1. id</li> <li>2. username</li> <li>3. password</li> </ol>	<ol style="list-style-type: none"> <li>1. id (Primary Key)</li> </ol>
surveys	<ol style="list-style-type: none"> <li>1. survey_id</li> <li>2. survey_name</li> <li>3. user_id</li> </ol>	<ol style="list-style-type: none"> <li>1. survey_id (Primary Key)</li> <li>2. user_id (Foreign Key)</li> </ol>
questions	<ol style="list-style-type: none"> <li>1. question_id</li> <li>2. question_description</li> <li>3. survey_id</li> <li>4. score</li> </ol>	<ol style="list-style-type: none"> <li>1. question_id (Primary Key)</li> <li>2. survey_id (Foreign Key)</li> </ol>
locations	<ol style="list-style-type: none"> <li>1. id</li> <li>2. location_description</li> </ol>	<ol style="list-style-type: none"> <li>1. id (Primary Key )</li> </ol>
agegroups	<ol style="list-style-type: none"> <li>1. id</li> <li>2. group_description</li> </ol>	<ol style="list-style-type: none"> <li>3. id (Primary Key )</li> </ol>
survey_agegroups	<ol style="list-style-type: none"> <li>1. id</li> <li>2. survey_id</li> <li>3. agegroup_id</li> </ol>	<ol style="list-style-type: none"> <li>1. id (Primary Key )</li> <li>2. survey_id (Foreign Key)</li> <li>3. agegroup_id (Foreign Key)</li> </ol>
genders	<ol style="list-style-type: none"> <li>1. id</li> <li>2. gender</li> </ol>	<ol style="list-style-type: none"> <li>1. id (Primary Key)</li> </ol>
survey_locations	<ol style="list-style-type: none"> <li>1. id</li> <li>2. survey_id</li> <li>3. location_id</li> </ol>	<ol style="list-style-type: none"> <li>1. id (Primary Key)</li> <li>2. survey_id (Foreign Key)</li> <li>3. location_id (Foreign Key)</li> </ol>

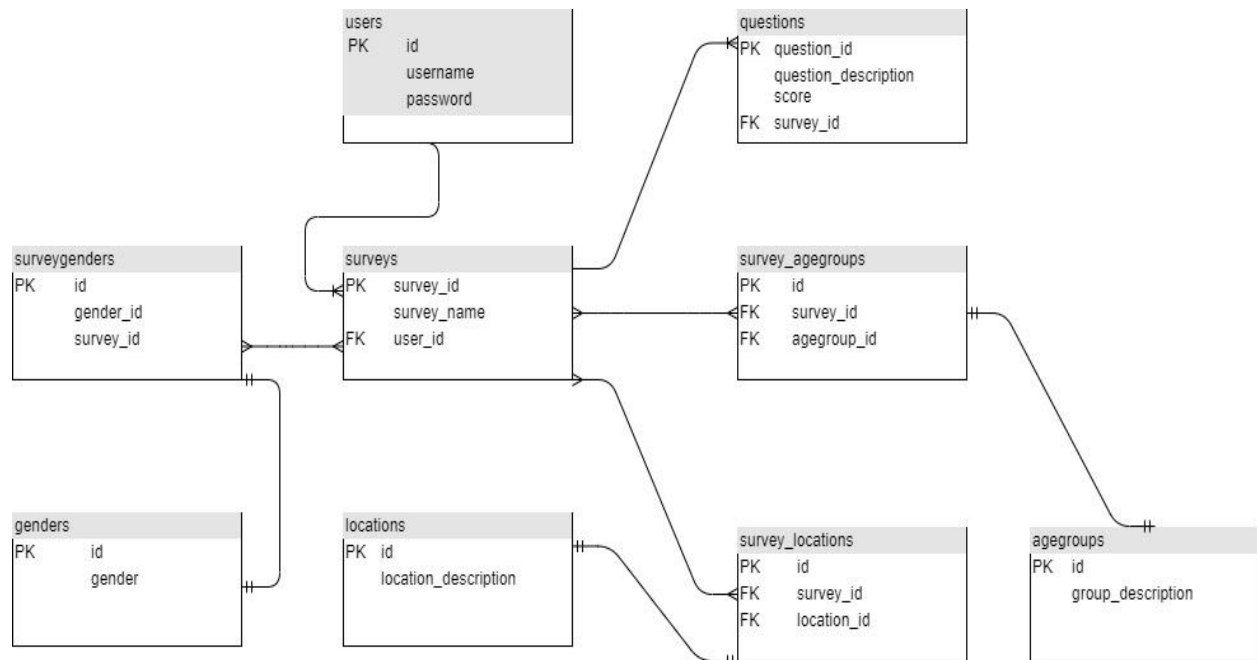


Figure : ER Diagram

## 3.1 Tables

### 3.1.1 users

users table is used to store the data of the administrator. Table has an id, username, and password, of which id is the primary key. The password and username from the table will be used to authenticate the user. After the user is authenticated he/she can create a survey.

### 3.1.2 surveys

surveys table is used to store the data of surveys created by the administrators. Surveys created by the user will be assigned a survey\_id. The table will store survey\_name, user\_id.

### 3.1.3 questions

questions table is used to store the questions received from the administrator. It will contain question\_description, survey\_id, and score of each question received from the customer. The score will be updated according to the response received from the customer.

### 3.1.4 locations

locations table stores the predefined location with their id. Customers can select their location from the list, which will update their location in the database.

### **3.1.5 age\_groups**

age\_groups table stores the predefined location with their id. Customers can select their age from the list, which will be updated in the database.

### **3.1.6 survey\_agegroups**

Survey\_agegroups table stores the agegroup\_id received from the customers along with survey\_id in the database.

### **3.1.7 survey\_locations**

survey\_locations table stores the location\_id received from the customers along with survey\_id in the database.

### **3.1.8 genders**

Gender table contains the predefined gender with their id. Customers can select their gender from the list, which will be updated in the database.

## **3.2 Endpoints**

### **3.2.1 UserRegister**

Registers administrators with their username and password.

- URL  
/register
- Method:  
POST
- URL Params Required:  
None
- Data Params:  
username=[string]  
password=[string]
- Success Response:  
Code=200

### 3.2.2 UserLogin

Users will be logged in after matching their username and password.

- URL  
/login
- Method:  
POST
- URL Params Required:  
None
- Data Params:  
username=[string]  
password=[string]
- Success Response:  
Code=200

### 3.2.3 AgeGroupList

Retrieves the age group list that are predefined and stored in age\_groups table.

- URL  
/agegroups
- Method:  
GET
- URL Params Required:  
None
- Data Params:  
None
- Success Response:  
Code=200

### 3.2.4 LocationList

Retrieves the location list that are predefined and stored in the locations table.

- URL  
/locations
- Method:  
GET
- URL Params Required:  
None

- Data Params:  
None
- Success Response:  
Code=200

### 3.2.5 Survey

Administrators can create surveys.

- URL  
`/create_survey/<string:username>`
- Method:  
POST
- URL Params Required:  
username
- Data Params:  
survey\_name=[string]  
questions=[dict]
- Success Response:  
Code=200

### 3.2.6 SurveyListByUsername

Administrator can retrieve the list of surveys he created.

- URL  
`/surveys`
- Method:  
GET
- URL Params Required:  
username
- Data Params:  
NONE
- Success Response:  
Code=200

### 3.2.7 SurveyList

User can retrieve the list of surveys.

- URL  
`/surveys`

- Method:  
GET
- URL Params Required:  
NONE
- Data Params:  
NONE
- Success Response:  
Code=200

### 3.2.8 QuestionList

Users can retrieve the list of questions from surveys.

- URL  
/surveys
- Method:  
GET
- URL Params Required:  
NONE
- Data Params:  
NONE
- Success Response:  
Code=200

### 3.2.9 Retrieve

Administrators can retrieve the responses received from customers for a particular survey. Response will consist of questions, updated score, location, and age group to which the customer belongs.

- URL  
/survey/<int:survey\_id>
- Method:  
GET
- URL Params Required:  
survey\_id
- Data Params:  
NONE
- Success Response:  
Code=200

### 3.2.10 UpdateSurvey

Customers can take the survey and each time the score associated with questions, location, and age group will be updated using this endpoint.

- URL  
`/submit_survey/<int:survey_id>`
- Method:  
`POST`
- URL Params Required:  
`survey_id`
- Data Params:  
`survey_name=[string]`  
`questions=[dict]`  
`location_id=[int]`  
`age_group_id=[int]`
- Success Response:  
`Code=200`

### 3.2.11 SurveyAgeGroupList

SurveyAgeGroupList stores and provides information about the age range of customers who have taken a particular survey.

- URL  
`/surveyagegroups`
- Method:  
`GET`
- URL Params Required:  
`NONE`
- Data Params:  
`NONE`
- Success Response:  
`Code=200`

### 3.2.12 SurveyLocationList

SurveyLocationList stores and provides information about the location range of customers who have taken a particular survey.

- URL  
`/surveylocations`



- Method:  
GET
- URL Params Required:  
NONE
- Data Params:  
NONE
- Success Response:  
Code=200

### 3.2.13 GenderList

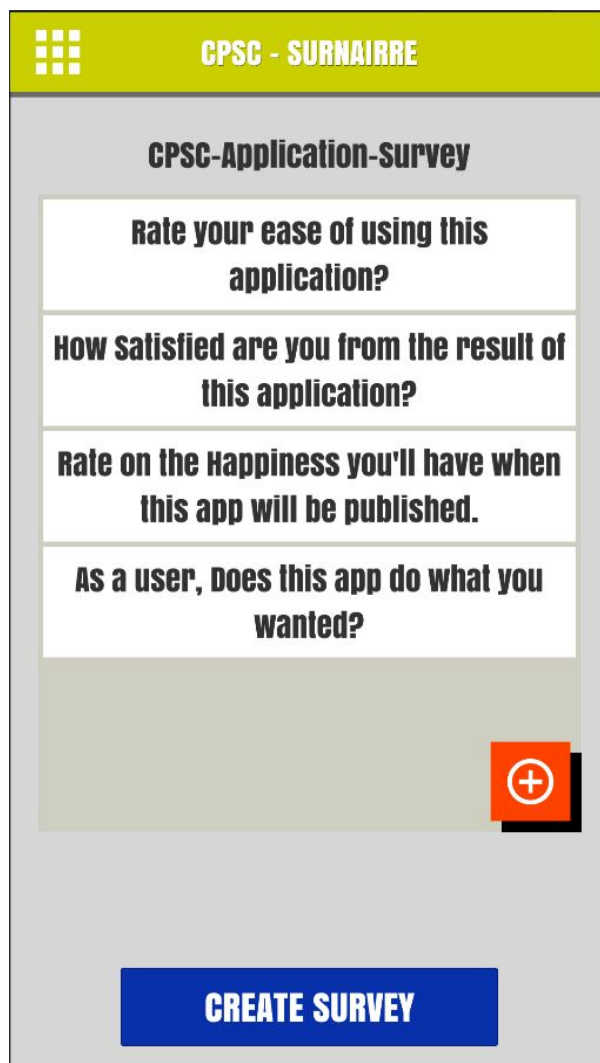
Retrieves the gender list that are predefined and stored in the locations table.

- URL  
/genders
- Method:  
GET
- URL Params Required:  
NONE
- Data Params:  
NONE
- Success Response:  
Code=200

## 4. Analysis

### 4.1 Survey Creation

To test the application we created a survey to get the feedback about the application. We asked 4 questions (each rating from 1- 10) to start with and ask our friends to participate in the survey. To keep things simple we decided to divide the locations into 3 regions i.e “India”, “Canada”, Others and Genders into “Male”, “Female”, Other. Age Group into “less than 20”, “20-24”, “41-60”, “greater than 60”.



The screenshot displays the 'Create Survey' interface. At the top, a yellow header bar contains a grid icon and the text 'CPSC - SURNAIRRE'. Below this, the title 'CPSC-Application-Survey' is centered. The main content area lists four survey questions, each in a white box with a light gray border: 'Rate your ease of using this application?', 'How satisfied are you from the result of this application?', 'Rate on the Happiness you'll have when this app will be published.', and 'As a user, Does this app do what you wanted?'. A large, empty light gray rectangular area is positioned below the questions, with a red square button featuring a white plus sign in its bottom right corner. At the very bottom, a blue button with the text 'CREATE SURVEY' is centered.

Figure : Create Survey Screen.

## 4.2 Result

As we can see from the images, a total 25 users participated in the survey with 13-males, 7-females, 5 Others. Age groups with age between 40-60 took max participation with 12 compared to other age-groups. Also we got 12 users from India, 10 from Canada and 3 from others locations. About the questions the second question i.e users are satisfied from the result of this application is the top scorer.

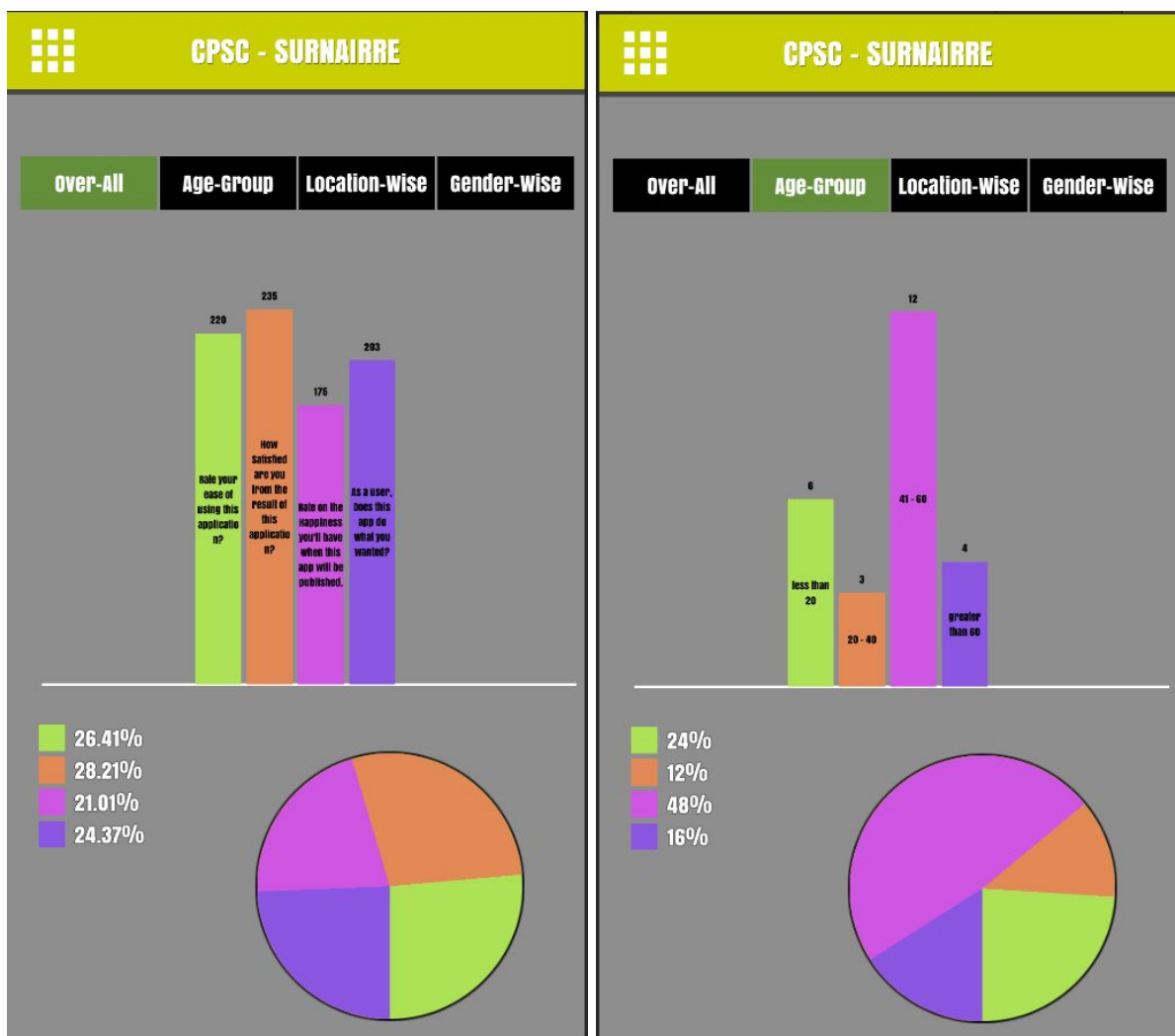


Figure : Over-all and Age-Group analysis Screen

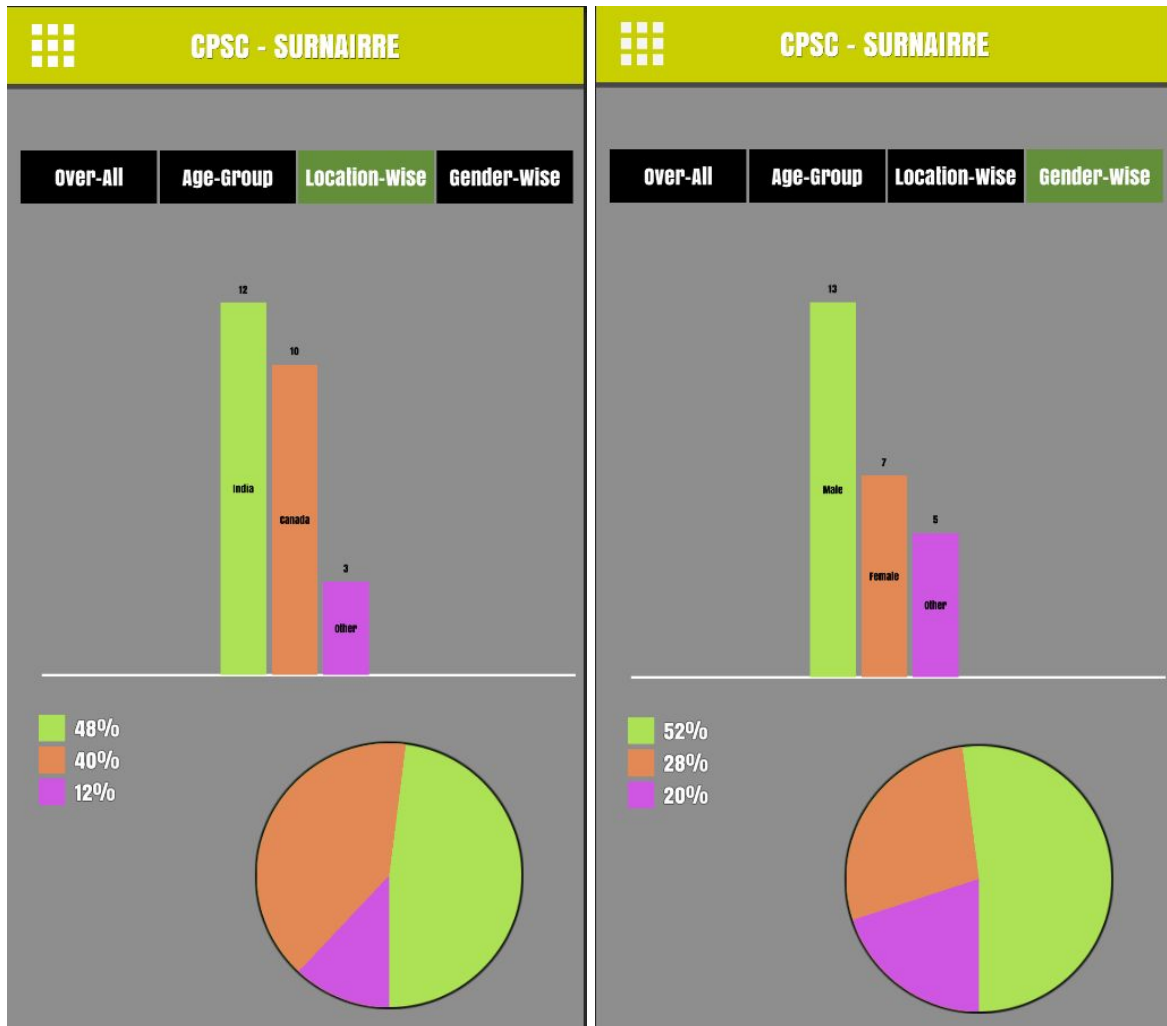


Figure : Locations-wise and Gender-wise analysis screens.

## **5. Conclusion**

Customer feedback is important and it can be used as a screening tool to identify topics of dissatisfaction. Good feedback generates personal recommendation and Bad feedback opens the door of improvements. Error or defects revealed in the satisfaction surveys or customer feedback procedure should lead to corrective or preventive actions, since customer feedback cannot result in quality improvement if proper actions are not carried out.

Furthermore, using surveys analysis one can find the proper audience of the product and the age groups, genders and can take the further step to increase the audience or target the different group of age, genders, location based on feedback received.

## References

- [1] <http://unity3d.com/>
- [2] <https://www.python.org/doc/>
- [3] <https://flask.palletsprojects.com/en/1.1.x/>
- [4] <https://docs.unity3d.com/ScriptReference/>
- [5] <https://docs.sqlalchemy.org/en/13/contents.html>
- [6] <https://learning.postman.com/docs/getting-started/introduction/>