

SRI GURU GOBIND SINGH **COLLEGE OF COMMERCE**

UNIVERSITY OF DELHI

PROJECT REPORT ON:

TRANCED: A CHATTING AND MUSIC PLAYER **APPLICATION**



SUBMITTED BY:

Anjali Garg

Garvit Bansal

Parmeet Singh

Harmanpreet Singh

GUIDANCE:

Ms. Musarrat Ahmed

SUBJECT:

Software Engineering

ACKNOWLEDGEMENT

We would like to express our sincere thanks to our Software Engineering teacher “Ms. Musarrat Ahmed” for giving us the opportunity to work under her on this project.

We truly appreciate and value her esteemed guidance and encouragement from the beginning to end of this project. We are extremely grateful to her. We want to thank all our teachers for providing a solid background for our studies. They have been great source of inspiration to us and we thank them from the bottom of our heart.

Last but not the least, we would like to thank our Computer Science department for providing us with all the facility that was required.

CERTIFICATE

This is to certify that Anjali Garg, Garvit Bansal, Parmeet Singh, Harmanpreet Singh students of B.Sc. (Hons) Computer Science Semester IV have submitted the project entitled “Tranced: A chatting and music player application” for the partial fulfilment of the requirements of Software Engineering project.

It embodies the work done by them during semester IV of their course under the due supervision of Ms. Musarrat Ahmed.

Ms. Musarrat Ahmed

INDEX

<u>S.No.</u>	<u>Topic</u>	<u>Page No.</u>
1.	Overview	1
2.	Problem Statement	2
3.	Advantages	3
4.	Process Model	4
5.	Initial Requirements	6
6.	Requirement Analysis	7
7.	Final Requirements	9
8.	Use Case Diagram	10
9.	Data Flow Diagrams	12
10.	Data Dictionary	14
11.	Sequence Diagrams	15
12.	Function Point Analysis	23
13.	Risk Management Plan	29
14.	Timeline Chart	31
15.	Pseudocode	32
16.	White Box Testing	34
17.	Interface Design	36

OVERVIEW

We all know that **music** and **communication** play an essential role in our life. Music has many benefits on human health and mood. It helps us to concentrate better and thus we perform things better. On the other hand, chatting is a method of using technology to bring people and ideas together despite of the geographical barriers.

Our project is an example of combining these two features into one application.

This android application will provide the following basic features to its users:

- A chat server which will enable its users to connect and chat with other users.
- A music server which will enable its users to stream online music while chatting.
- An interacting interface to allow its users to easily explore and use all the features provided by the application

PROBLEM STATEMENT

The joy that comes from listening to music with friends is something that's been around ever since...well, ever since music. But today, thanks to digital services that let you listen anytime, anywhere, music is now more often a solo experience. A new app called **TRANCED** wants to change that, by offering you a way to listen to music with a friend, no matter how far apart you are, and chat about it right in the app.

This project overcomes the drawback of apps like YouTube and WhatsApp i.e., one can listen to songs while chatting.

"It's like sharing earphones," and indeed, it does seem like the modern-day version of handing one of your earbuds to a friend.

ADVANTAGES

USP (Unique Selling Proposition)

We have numerous music player and chatting applications available but none of them gives us the facility to share and listen our favourite songs with our friends in the same app. Well, now you have TRANCED application for that.

Some other advantages provided by this application are stated as:

➤ Accurate sizing of the requirements:

This application has a modular structure. Almost all components of the system are equipped with a basic functionality, which can be extended with additional modules according to user demand.

➤ Low installation cost:

Time is money. For this reason, great value is placed on a simple and quick installation of the component i.e., this application does not need any additional hardware for its functioning. It can work perfectly on existing system.

➤ Speedy System, smooth functioning:

With this application user will be able to listen music and connect with other users simultaneously and smoothly.

➤ Proper utilization of advanced technology:

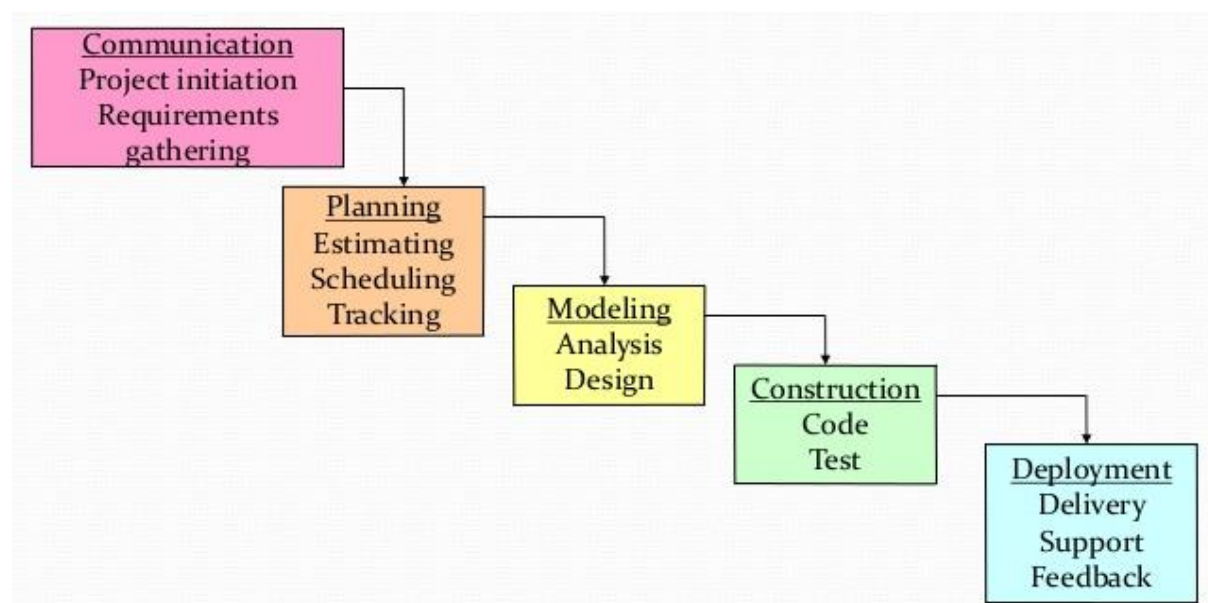
The project promotes use of advanced and modern technologies for betterment of society and advancement of existing facilities.

PROCESS MODEL

A software process model represents the order in which the activities of software development will be undertaken. It describes the sequence in which the phases of the software lifecycle will be performed.

The model that we are using to develop our project is **Linear Sequential Model**. This model is sometimes called the **Classic Life Cycle** or **Waterfall Model**.

The linear sequential model suggests a systematic, sequential approach to software development that begins with customer specification of requirements and progresses through planning, modelling, construction, and deployment culminating in on-going support of the completed software.



We are opting for the Linear Sequential Model because:

- All requirements for the project have been explicitly stated at the beginning. There is very little scope of customer's deviation from current requirements, coding and testing after detailed analysis is much easy.
- The requirements are not very complex in nature. Hence, the system has low complexity.

- The software works in a linear fashion and there is little scope of changing the system once it has been setup.

Different phases of the model:

1.**Software requirements analysis:** In this, software engineer understands the nature of a program to be built, he/she must understand the information domain for the software as well as required function, behaviour, performance and interface. Requirements for both the system and the software are documented and reviewed with the customer.

- Communication
- Planning
- Modelling
- Construction
- Deployment

2.**Design:** It has four distinct attributes of a program: data structure, software architecture, interface representation and procedural details. It is documented and becomes part of the software.

3.**Code generation:** Design must be translated into a machine-readable form which is done by code generation.

4.**Testing:** It focuses on the logical internals of the software, ensuring that all the statements have been tested, and on the functional externals; that is conducting test to uncover errors and ensure that defined input will produce actual results.

5.**Support:** This is a phase when software will undoubtedly undergo change after it is delivered to the customer. Change will occur because errors have been encountered, and the software must be adapted to accommodate changes in its external environment, or because the customer requires functional or performance enhancements. Software support/maintenance reapplies each of the preceding phrases to an existing program rather than a new one.

INITIAL REQUIREMENTS

User Sign-Up: New User will have to Sign-Up before using the application.

User Sign-In: User will Sign-In and then he/she can use the application to listen to songs or chat with friends.

Search bar for songs: User can search songs using song names.

Personal playlist: User can save songs by marking it as favorite and can access their customized playlist later.

Recommendation: Besides the playlist made by user, the application will recommend playlists and songs.

Chat feature: User can chat with their friends in the chat tab.

Friend Request: User can make friends by sending them friend request.

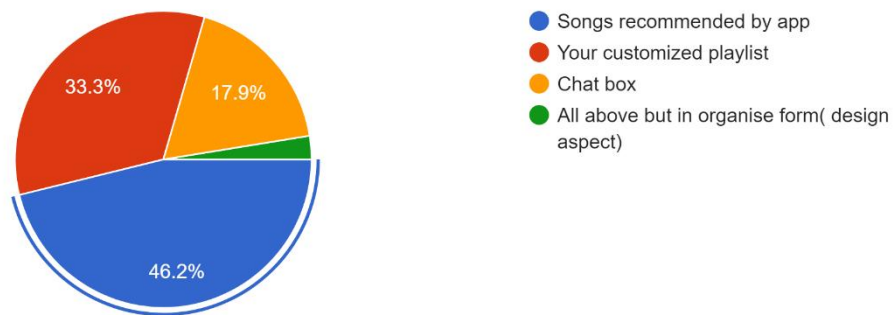
REQUIREMENT ANALYSIS

A survey was conducted with friends and families to have a better understanding on people's expectations and opinion on our project.

Here's the response:

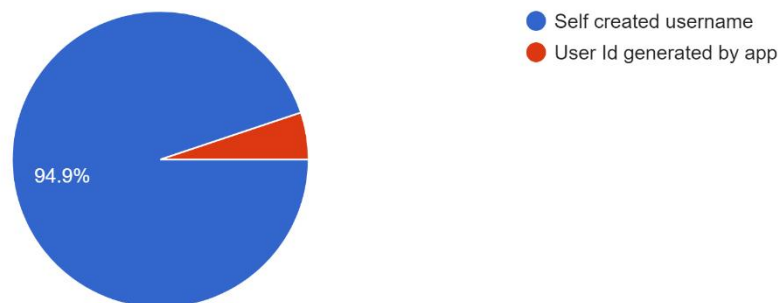
What feature would you like to see on the home screen?

39 responses



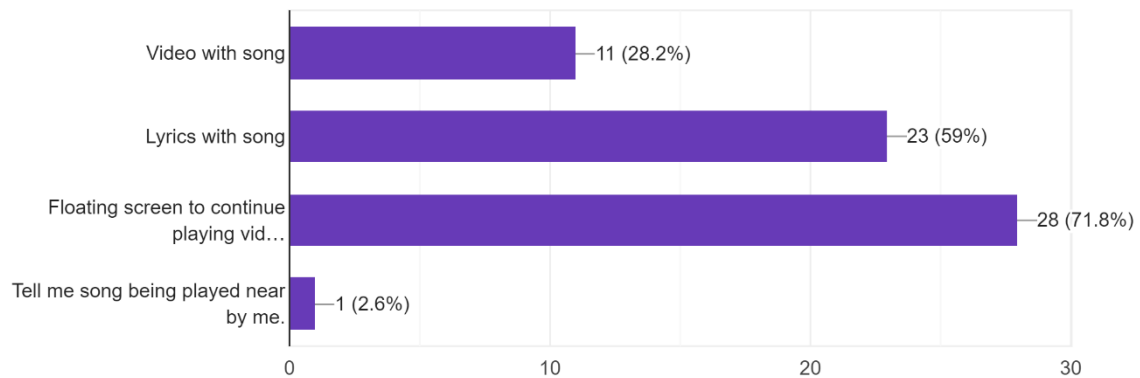
How would you prefer to connect with your friends?

39 responses



What other features would you like us to add?

39 responses



Would you like us to add or remove any feature?

19 responses

genre of song user usually listen...

All good

Good Luck from Yuvraj

1) Add feature that tells me the song playing near by me like Google assistant does

2) Add mood feature- that automatically play songs according to my mood like party,romance or travel.

3) Please Remove excessive ads .

add:-shows gender

Add language options if possible.

Kyu btau

Add classic old songs

Filter by Artist

FINIAL REQUIREMENTS

User Login/ Sign-up:

Firstly, one has to register themselves to server using their Gmail or Facebook Account to avail the chat feature and can also listen to offline music through phone, to stream online music one must link their account to Spotify server.

Search Bar:

One can avail the search option to find their favourite songs, albums, can also search songs by artist or movie name.

Personal Playlists:

One can also create their own playlists simply by adding songs from server to their favourites or they can also create new playlists according to their mood.

Recommendation:

Recommended songs will also be shown by the application itself on the home screen

Friend Request:

User can make friends by sending them friend request using self-created usernames.

Chat Feature and Floating Window:

One can also chat with their friends using in built chat feature provided by application, one can chat while watching video songs.

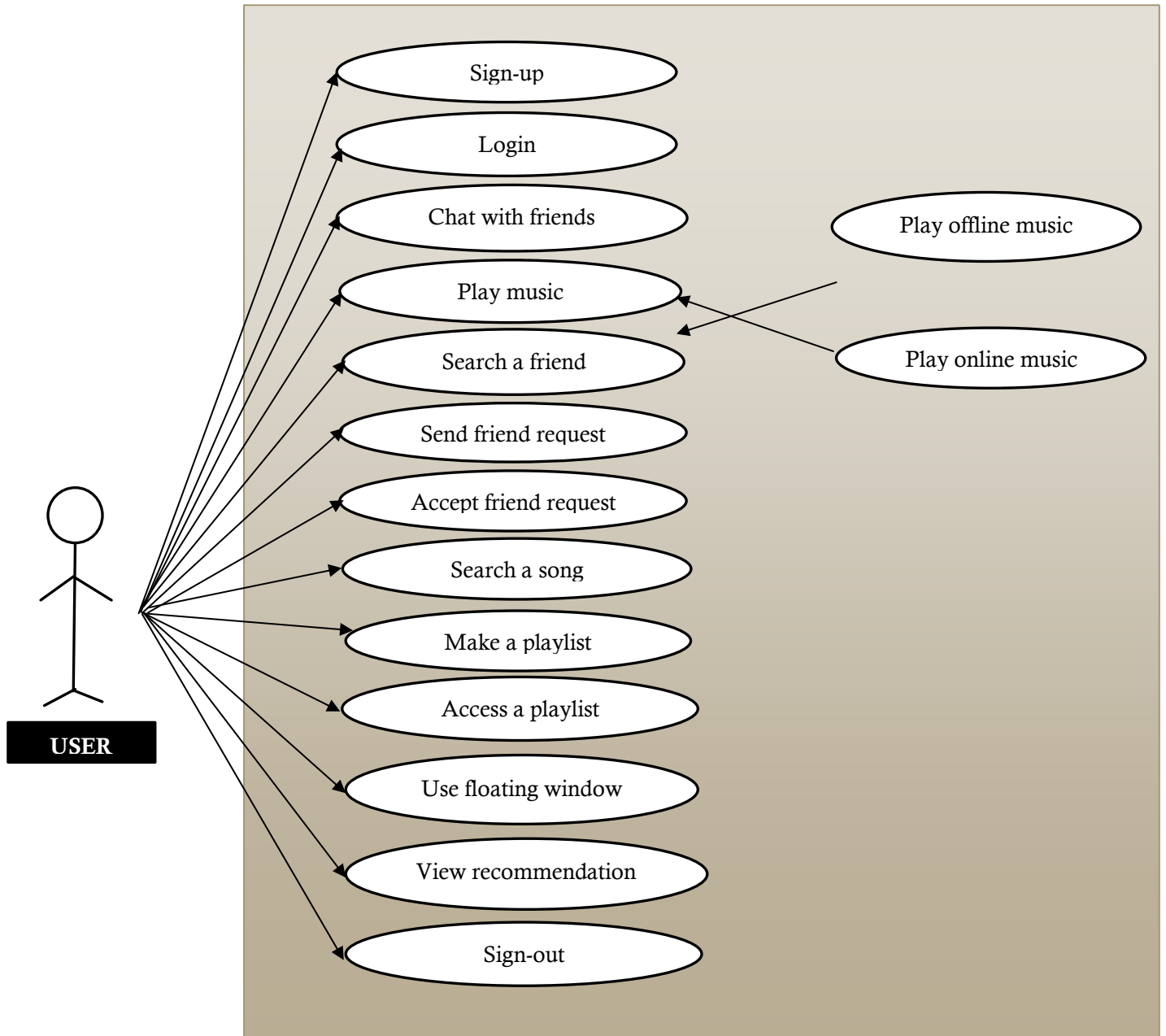
USE CASE DIAGRAM

Use case diagrams are considered for high level requirement analysis of a system. When the requirements of a system are analysed, the functionalities are captured in use cases. Use cases are used to show system functionalities written in an organized manner. The second thing which is relevant to use cases are the actors. Actors can be defined as something that interacts with the system.

The purposes of use case diagrams can be said to be as follows:

- Used to gather the requirements of a system.
- Used to get an outside view of a system.
- Identify the external and internal factors influencing the system.
- Show the interaction among the requirements and actors.

USE CASE DIAGRAM FOR TRANCED

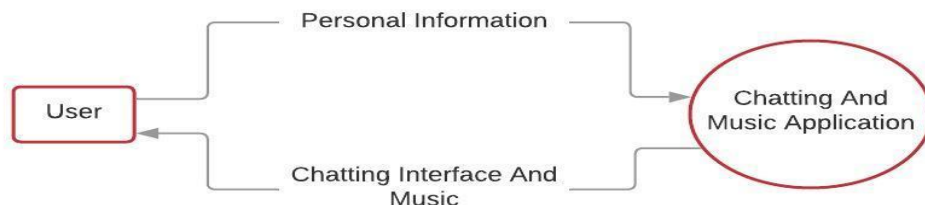


DATA FLOW DIAGRAM(DFD)

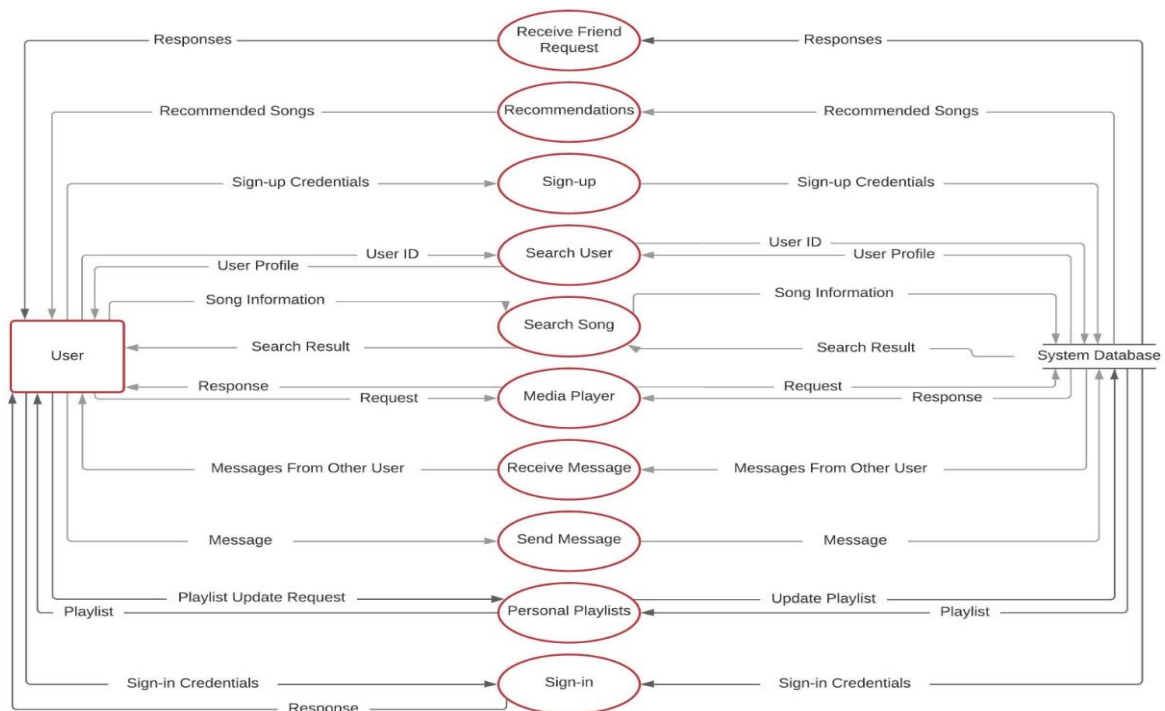
A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated.

The benefits of data-flow diagrams: Data-flow diagrams provide a very important tool for software engineering, for a number of reasons: The system scope and boundaries are clearly indicated on the diagrams.

LEVEL 0 DFD (Context Diagram)

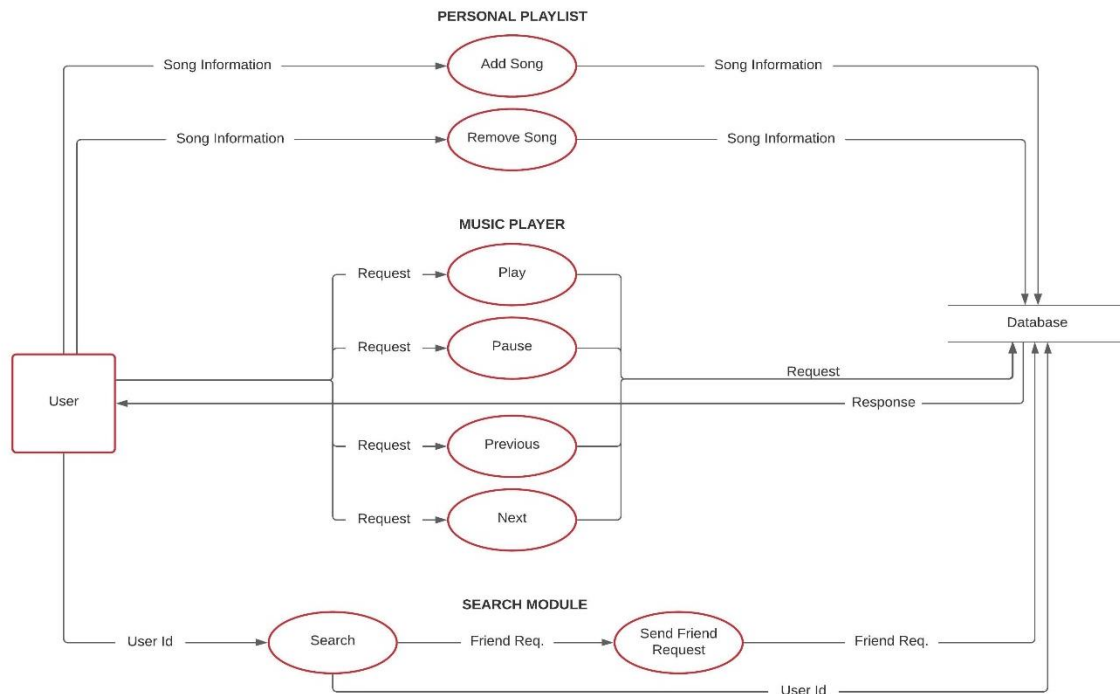


LEVEL 1 DFD



LEVEL 2 DFD

We expanded the functions: Search User, Personal Playlist and Media Player from level 1 DFD.



DATA DICTIONARY

Sign-up Credentials: [Profile picture] + Email + Username + Password + Confirm Password

Sign-in Credentials: Email + Password

User Id: Username

User Profile: Username + Send Request | Receive Request + [User's Bio]

Song information: Song Name | Artist Name | Album Name

Playback Screen Options: Song Name + Add to favourites option + Media Player Options

Media Player Options: Play | Pause + [Previous Song | Next Song]

Update Playlist Option: Add song + Remove song

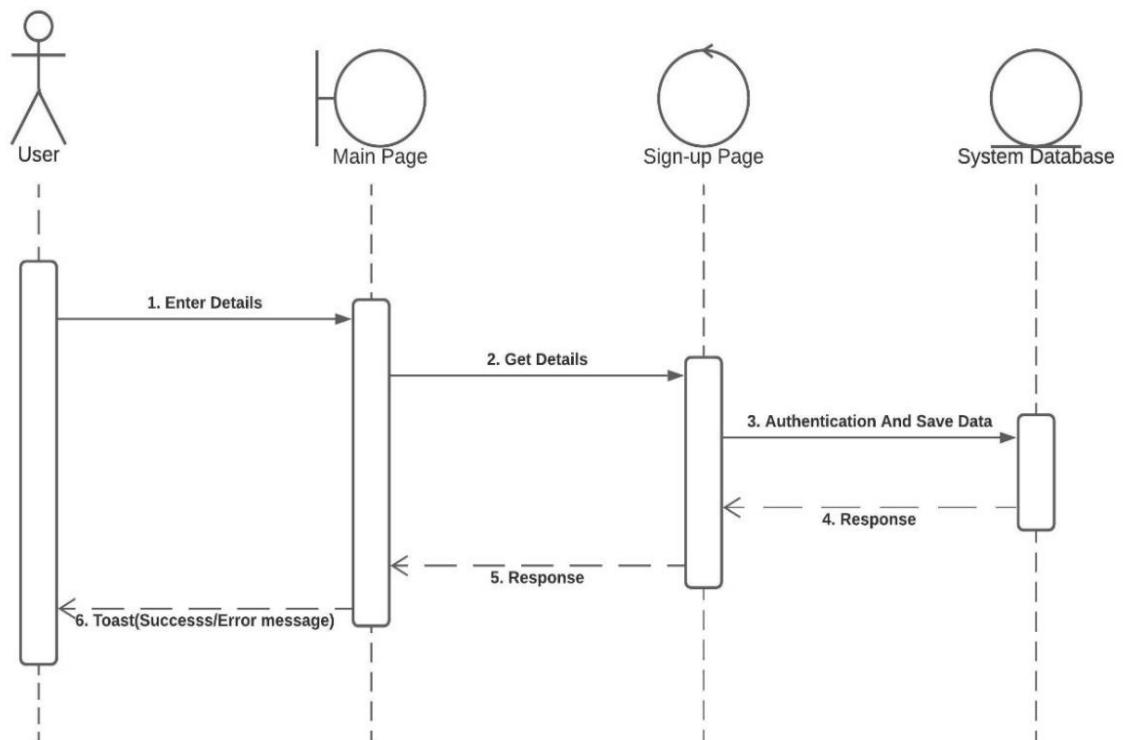
Search Friend Option: Users Profile + Send Request Option

Friend Request Option: All pending requests + Response Option

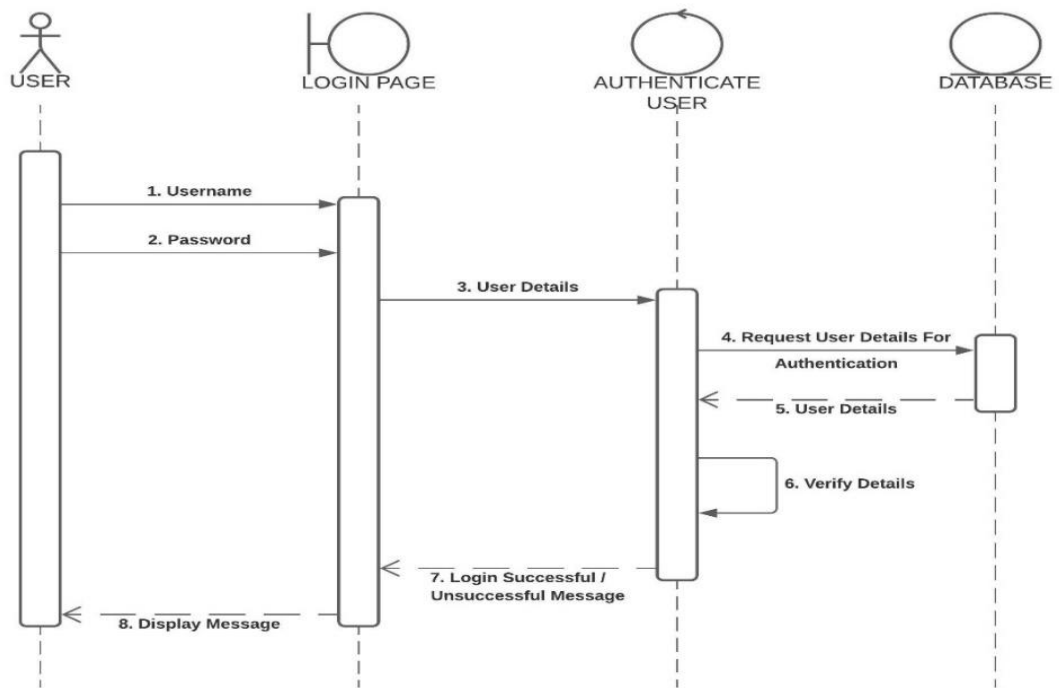
SEQUENCE DIAGRAMS:

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time and horizontal arrows to show what actions are taken and when.

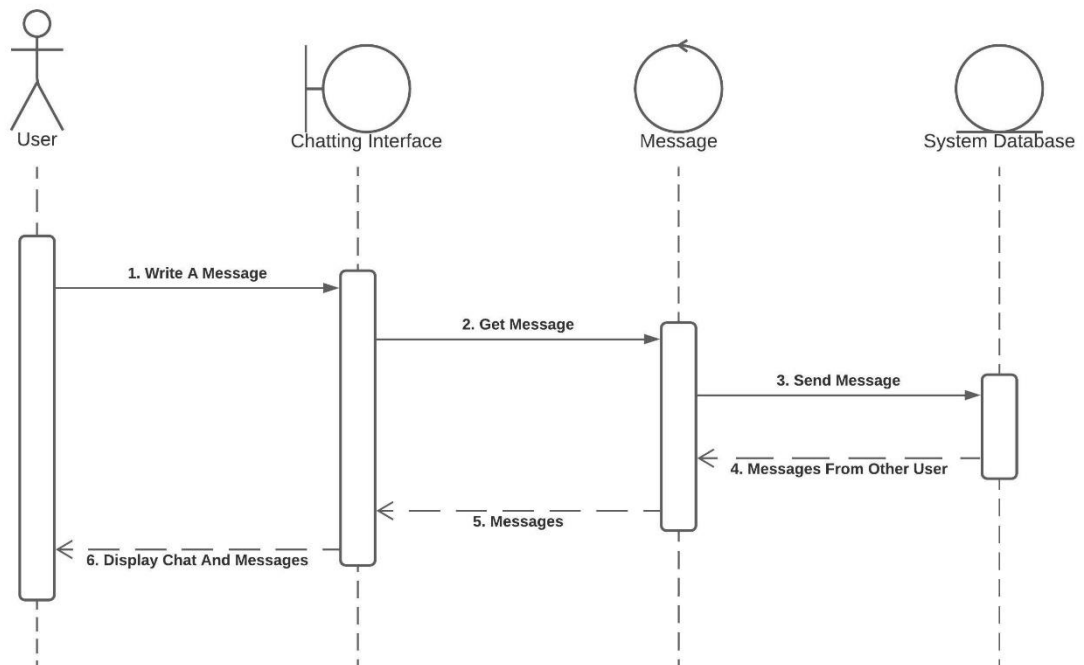
Sequence diagram for use case 'Sign-up'



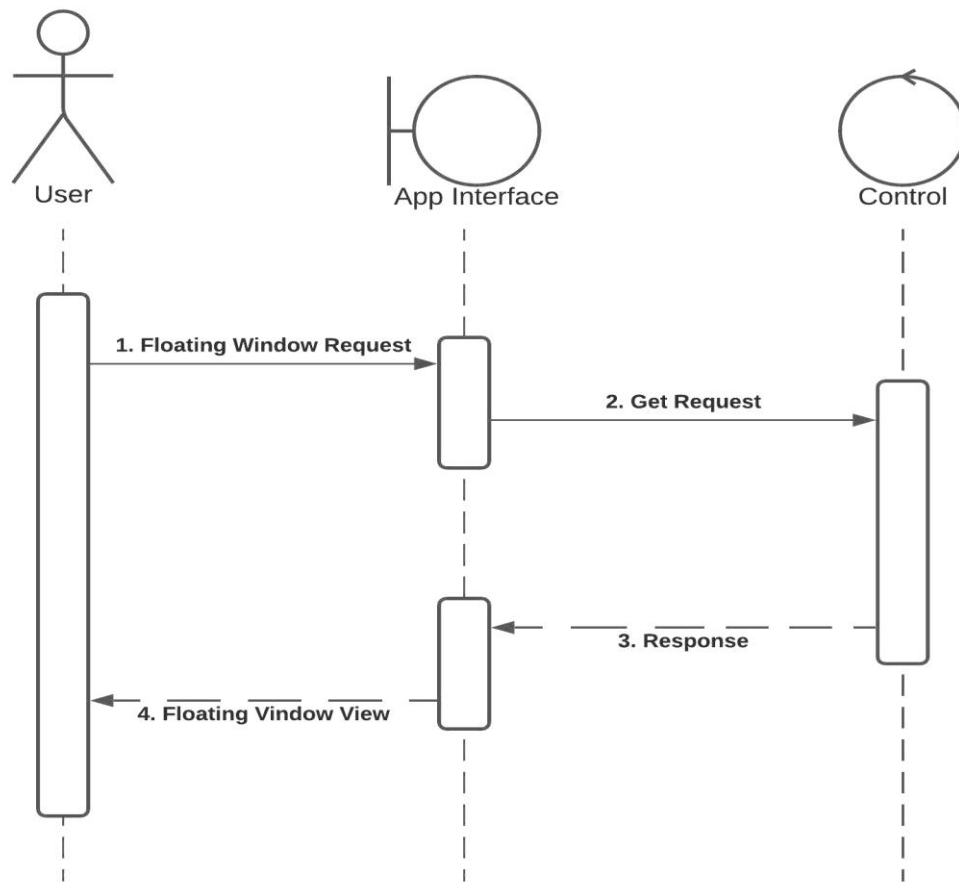
Sequence diagram for use case 'Login'



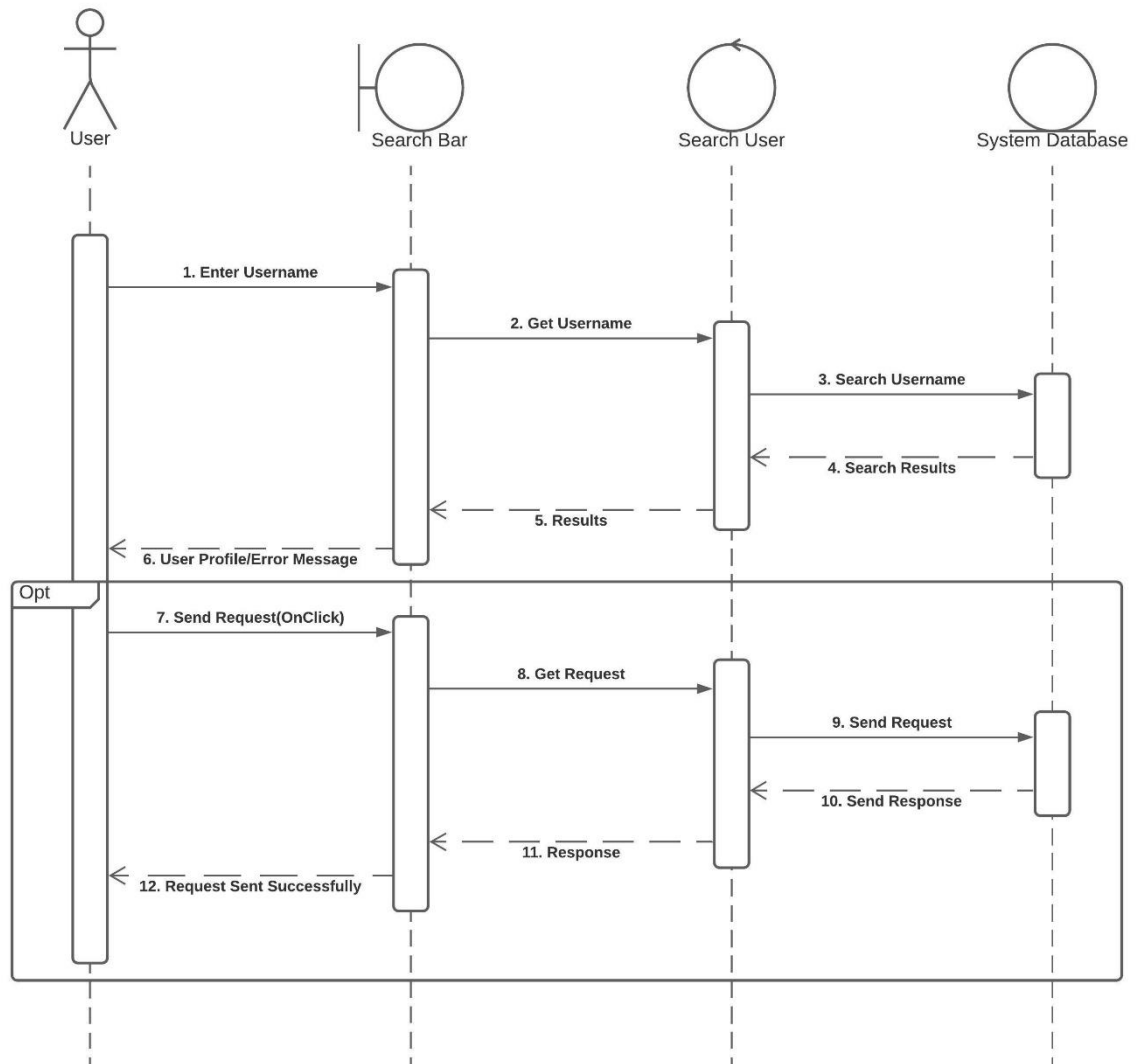
Sequence diagram for use case 'Chat with friends'



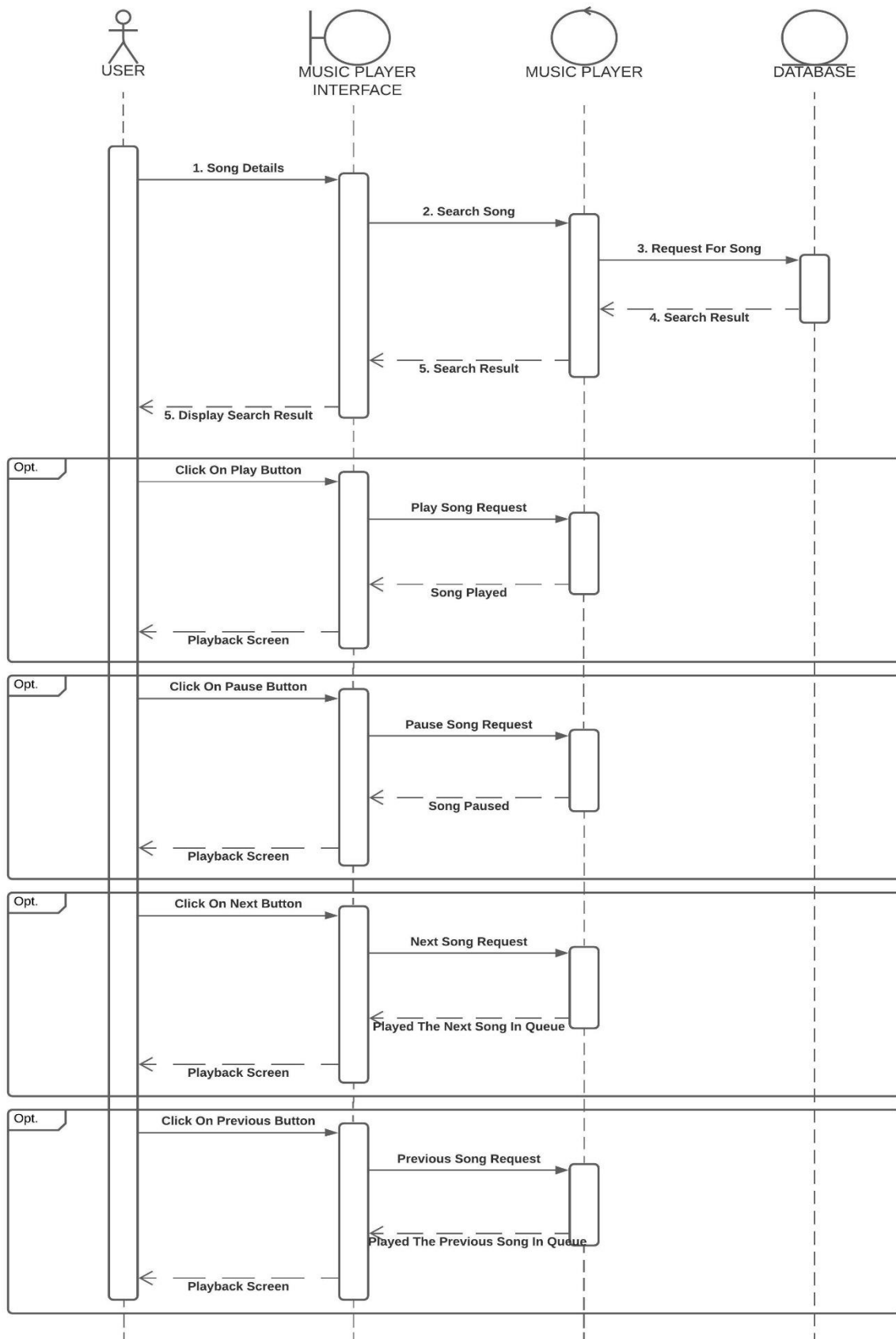
Sequence diagram for use case 'Floating Window'



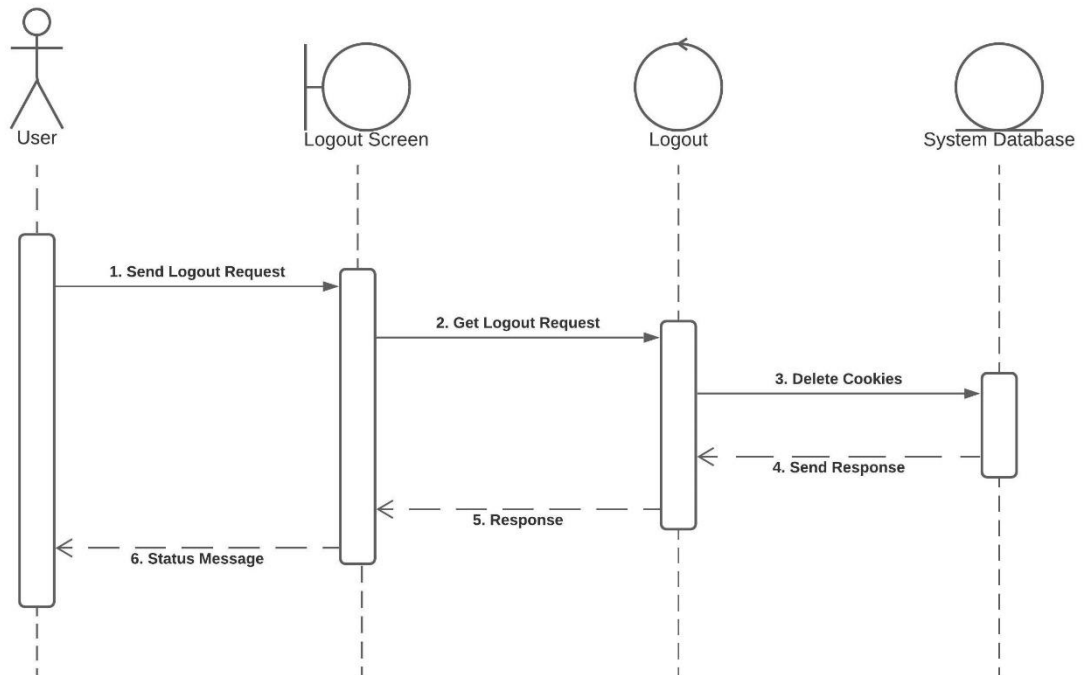
Sequence diagram for use case 'Search User'



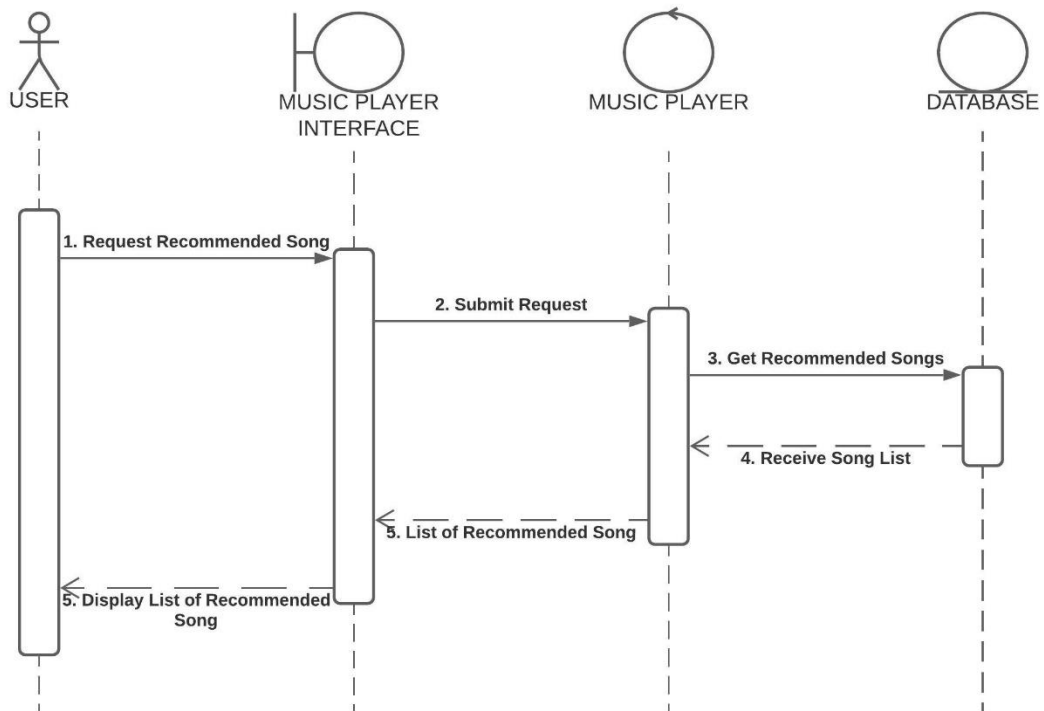
Sequence diagram for use case 'Play Music' & 'Search a Song'



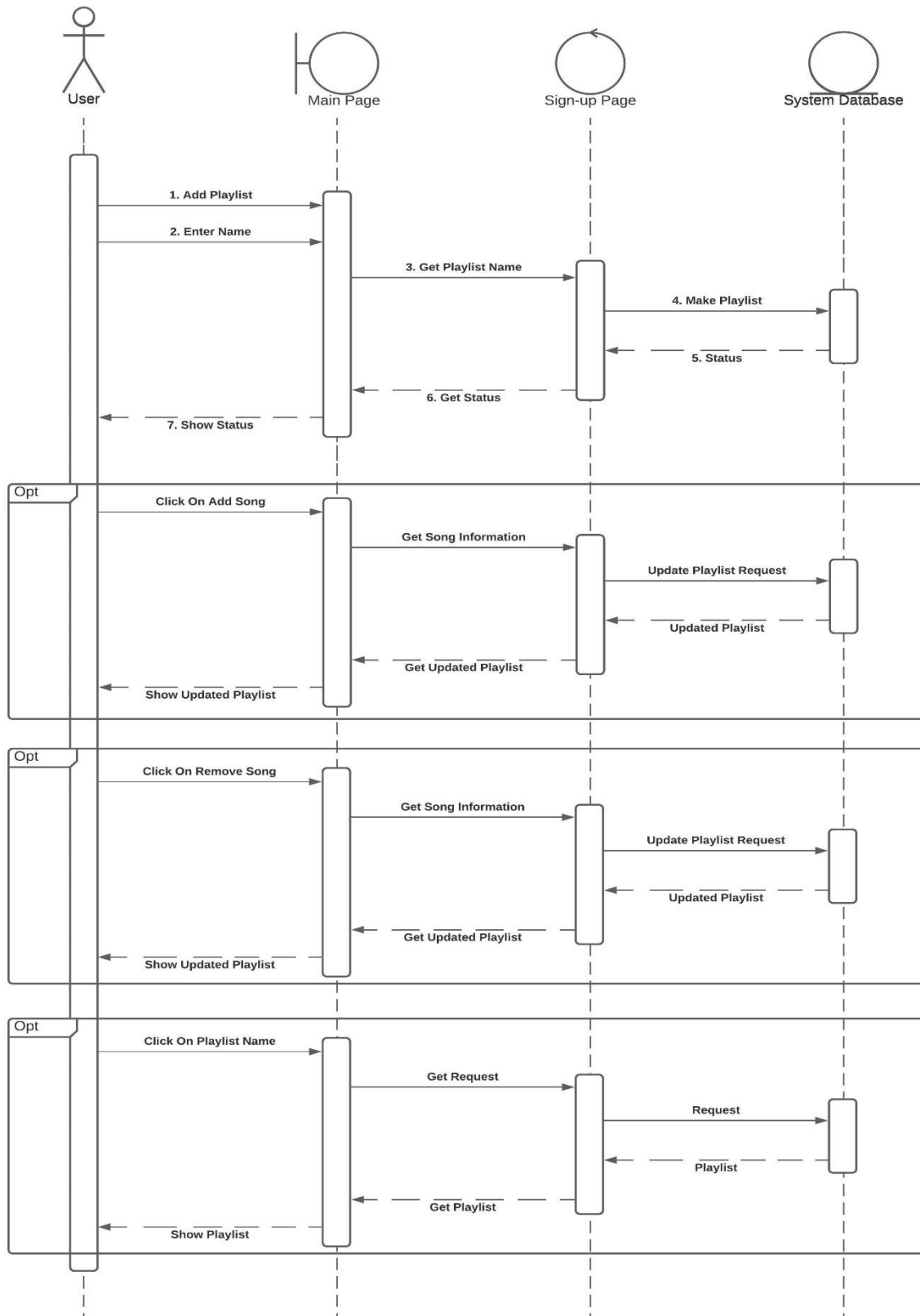
Sequence diagram for use case 'Logout'



Sequence diagram for use case 'Recommendations'



Sequence diagram for use case 'Make a playlist'



FUNCTION POINT ANALYSIS

FPA is a standard metric for the relative size and complexity of a software system, originally developed by Alan Albrecht of IBM in the late 1970s.

Function Points (FPs) can be used to estimate the relative size and complexity of software in the early stages of development- analysis and design.

It assesses the functionality delivered to its users, based on the user's external view of the functional requirements. It measures the logical view of an application not the physically implemented view or the internal technical view.

The FPA technique is used to analyse the functionality delivered by software and *Unadjusted Function Point (UFP)* is the unit of measurement.

Objectives of FPA:

1. The objective of FPA is to measure functionality that the user requests and receives.
2. The objective of FPA is to measure software development and maintenance independently of technology used for implementation.
3. It should be simple enough to minimize the overhead of the measurement process.
4. It should be a consistent measure among various projects and organizations.

Types of FPA:

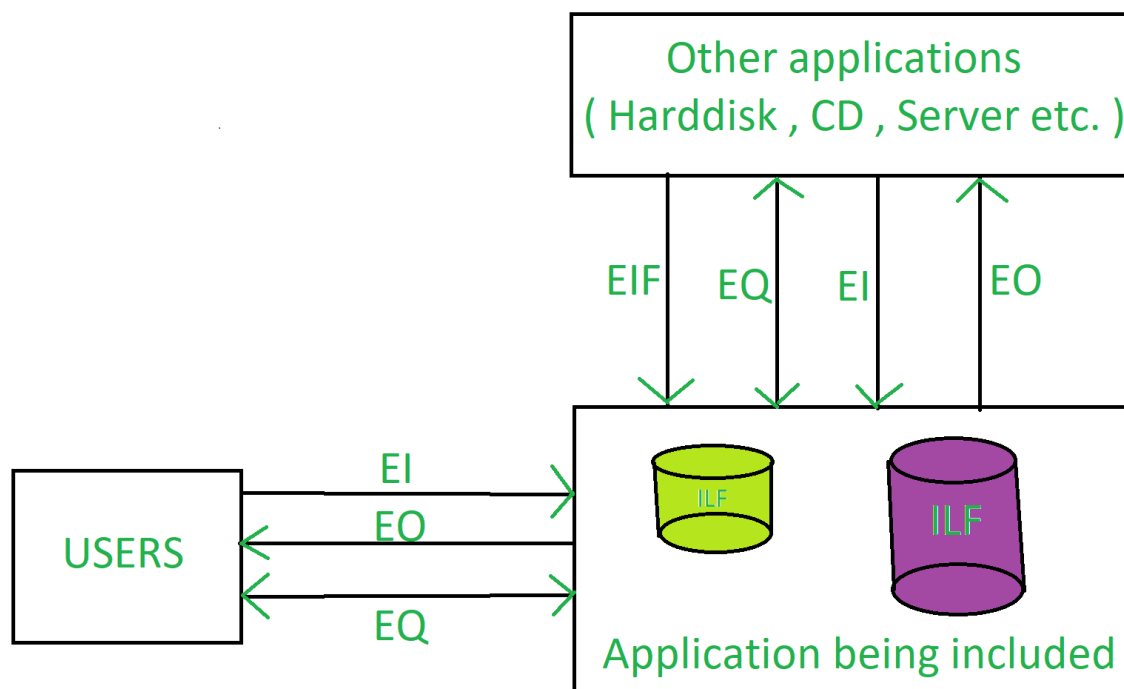
Transactional Functional Type –

1. External Input (EI): EI processes data or control information that comes from outside the application's boundary. The EI is an elementary process.
2. External Output (EO): EO is an elementary process that generates data or control information sent outside the application's boundary.

3. External Inquiries (EQ): EQ is an elementary process made up of an input-output combination that results in data retrieval.

Data Functional Type –

4. Internal Logical File (ILF): A user identifiable group of logically related data or control information maintained within the boundary of the application.
5. External Interface File (EIF): A group of user recognizable logically related data allusion to the software but maintained within the boundary of another software.



Benefits of FPA:

- FPA is a tool to determine the size of a purchased application package by counting all the functions included in the package.
- It is a tool to help users discover the benefit of an application package to their organization by counting functions that specifically match their requirements.
- It is a tool to measure the units of a software product to support quality and productivity analysis.

STEP1: ADJUSTMENT VALUES TABLE

<u>S.No.</u>	<u>Parameters</u>	<u>Adjustment Factor</u>
1.	Does the system require reliable backup and recovery?	f1=3
2.	Are specialized data communications required to transfer information to or from application?	f2=4
3.	Are there distributed processing functions?	f3=3
4.	Is performance critical?	f4=4
5.	Will the system run in an existing, heavily utilized operational Environment?	f5=4
6.	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	f6=3
7.	Does the system require online data entry?	f7=4
8.	Is the code designed to be reusable?	f8=3
9.	Is the system designed for multiple installations in different organizations?	f9=3
10.	Are the inputs, outputs, inquiries complex?	f10=1
11.	Are the ILFS updated online?	f11=2
12.	Is the internal processing complex?	f12=3
13.	Are conversion and installation included in the design?	f13=1
14.	Is the application designed to facilitate change and ease of use by the user?	f14=4

Scale varies from 0 to 5 according to character of Complexity Adjustment Factor (CAF). Below table shows scale:

0 = No influence

1 = Incidental

2 = Moderate

3 = Average

4 = Significant

5 = Essential

Formula for calculating Complexity Adjustment Factor (CAF):

$$\text{CAF} = 0.65 + 0.01 * \sum f_i$$

Calculating $\sum f_i$

$$\begin{aligned}\sum f_i &= f_1 + f_2 + f_3 + f_4 + f_5 + f_6 + f_7 + f_8 + f_9 + f_{10} + f_{11} + f_{12} + f_{13} + f_{14} \\ &= 3 + 4 + 3 + 4 + 4 + 3 + 4 + 3 + 3 + 1 + 2 + 3 + 1 + 4 \\ &= 42\end{aligned}$$

Calculating CAF

$$\begin{aligned}\text{CAF} &= 0.65 + 0.01 * \sum f_i \\ &= 0.65 + 0.01 * 42 \\ &= 0.65 + 0.42 \\ &= 1.07\end{aligned}$$

STEP 2: UNADJUSTED FUNCTION POINT (UFP)

EXTERNAL INPUTS:

Input Name	Fields
Sign-up	4
Login	2
Send Message	1
Search Song	1
Search User	1
Send Request	1

EXTERNAL OUTPUTS:

Output Name	Fields
Search Song	3
Search User	1
Playlist	1

EXTERNAL INQUIRIES:

Inquiries	Fields
Chat	1
Playlist	1
Media Player	4
YouTube Player	1
New Release	1
User Info	4
Recommended Songs	1
Friend Requests	1

INTERNAL LOGICAL FILES:

File Name	Fields
User Details	4
Profile Image	1
Artists	1
Albums	1
Tracks	1
Playlists	1
Recommendations	1

EXTERNAL INTERFACE FILES:

File Name	Fields
Profile Image	1
Documents	1

TABLE FOR UNADJUSTED FUNCTION POINT (UFP):

Calculation:

The weight factor is assumed to be average.

<u>Function Type</u>	<u>Estimated Factor</u>	<u>Weight Factor</u>			<u>Function Type Total</u>
		<u>Low</u>	<u>Average</u>	<u>High</u>	
EI	6	3	4	6	$6*4 = 24$
EO	3	4	5	7	$3*5 = 15$
EQ	8	3	4	6	$8*4 = 32$
ILF	7	7	10	15	$7*10 = 70$
EIF	2	5	7	10	$2*7 = 14$

Total Unadjusted Function Point Count = $24 + 15 + 32 + 70 + 14$
= 155

Calculating Function Point Count

$AFP = UFP * CAF$

$AFP = 155 * 1.07$

AFP = 165.85

Where AFP = Adjusted Function Point

UFP = Unadjusted Function Point

CAF = Complexity Adjustment Factor

RISK MANAGEMENT PLAN

A software project can be concerned with a large variety of risks. In order to be adept to systematically identify the significant risks which might affect a software project, it is essential to classify risks into different classes. The project manager can then check which risks from each class are relevant to the project.

There are three main classifications of risks which can affect a software project:

- Project risks
- Technical risks
- Business risks

1. Project risks: Project risks concern different forms of budgetary, schedule, personnel, resource, and customer-related problems. A vital project risk is schedule slippage. Since the software is intangible, it is very tough to monitor and control a software project.

2. Technical risks: Technical risks concern potential method, implementation, interfacing, testing, and maintenance issue. It also consists of an ambiguous specification, incomplete specification, changing specification, technical uncertainty, and technical obsolescence. Most technical risks appear due to the development team's insufficient knowledge about the project.

3. Business risks: Business risks threaten the viability of the software to be built and often jeopardise the project or the product. Candidates for the top five business risks are:

- Market risk: Building an excellent product or system that no one really wants.
- Strategic risk: Building a product that no longer fits into the overall business strategy for the company.
- Sales risk: building a product that the sales force doesn't understand how to sell.

- Management risk: Losing the support of senior management due to a change in focus or a change in people.
- Budget risk: losing budgetary or personnel commitment.

RMMM TABLE

Impact Rates:

1.Catastrophic 2. Critical 3. Marginal 4. Negligible

<u>S. No.</u>	<u>Risk</u>	<u>Category</u>	<u>Probability</u>	<u>Impact</u>	<u>RMMM</u>
1.	The team may lose all the project artifacts any time during the project and thus will be unable to deliver the application to the customer. Such an unlikely event may be caused by a hard disk failure , etc.	Technical Risk	5%	1	To avoid losing the work already completed, the team will have to carry out a necessary backup of database data, source, code and documentation. Ensure that backups are made in regular intervals of time.
2.	Customer requirements might change. Since our software and system is made in a linear fashion, changing of requirements can be a big problem.	Project Risk	20%	2	SRS should be documented and validated with the customer in advance.
3.	Delivery deadline may be tightened.	Business Risk	20%	2	Schedule made should be realistic and achievable. Monitor that efforts put are according to schedule.
4.	Lack of training on tools or insufficient skills for operating the system.	Technical Risk	10%	3	Staff must be trained to Handle working of tools.
5.	Team dissension/ Lack cohesion	Project Risk	10%	4	Some guidelines and rules should be set describing how to deal with each other.

TIMELINE CHART

Project management is a balancing act, and project managers need project management skills and tools to keep every ball up in the air. And then there's the timeline chart that illustrates the project's schedule to keep the project on track.

Timeline charts illustrate project task or activity duration, start and end dates, the time between tasks, any task overlap, and the dependencies between tasks, if any. And it's a great resource to detail and highlight your work in summary reports for stakeholders.

<u>S. No.</u>	<u>PROCESS/PHASE</u>	<u>START DATE</u>	<u>END DATE</u>
1.	Process Model	17.01.21	22.01.21
2.	Requirement Analysis	4.02.21	11.02.21
3.	Use Case Diagrams	17.02.21	25.02.21
4.	Data Flow Diagrams	27.02.21	1.02.21
5.	Sequence Diagrams	11.03.21	16.03.21
6.	Function Point Analysis	26.03.21	6.04.21
7.	Risk Management Plan	13.04.21	15.04.21
8.	Pseudocode	18.04.21	20.04.21
9.	White Box Testing	23.04.21	27.04.21
10.	Design	19.04.21	28.04.21

PSEUDOCODE

Search Module

```
onCreate(Bundle savedInstanceState) {  
    MediaPlayer mediaPlayer = new MediaPlayer  
    mediaPlayer.setAudioAttributes((AudioAttributes) object)  
    RecyclerView R1 <- link to (R.id.R1)    //RecyclerView view for tracks  
    RecyclerView R2 <- link to (R.id.R2)    //RecyclerView view for albums  
(horizontal layout)  
    RecyclerView R3 <- link to (R.id.R3)    //RecyclerView view for artists (horizontal  
layout)  
    Intent I = get intent from previous activity()  
    String m_Key = String from intent with key("key")    //Stored Auth token  
from previous activity  
    String q = String from intent with key ("query") //Stored query from previous  
activity  
    Call<SearchTra> tracks = new SearchService() object call to  
searchtrack(q,"track","IN")  
    tracks.enqueue(Callback<SearchTra> object)  
    {  
        onSuccess{ Adding adapter on success in R1}  
        onError{}  
    }  
    Call<SearchAlb> tracks = new SearchService() object call to  
searchtrack(q,"album","IN")  
    {  
        onSuccess{ Adding adapter on success in R2}  
        onError{}  
    }  
  
    tracks.enqueue(Callback<SearchAlb> object)  
    Call<SearchArt> tracks = new SearchService() object call to  
searchtrack(q,"artist","IN")  
    tracks.enqueue(Callback<SearchArt> object)  
    {  
        onSuccess{ Adding adapter on success in R3}  
        onError{}  
    }  
}
```

//Here Call<T> method is for retro fit call for hitting Endpoints

//Functions used above in call request is as follows they are member functions of class GetService:-

```
public static get_item get(String token){
    if(GI == null)
    {
        OkHttpClient client = new
OkHttpClient.Builder().addInterceptor(new Interceptor() {
            @Override
            public Response intercept(Chain chain) throws
IOException {
                Request newRequest = new request with header
(Authentication Bearer token)
                return chain.proceed(newRequest);
            }
        }).build();

        Retrofit retrofit = new retrofit object build by using gson
factory and endpoint
                                and client defined above
        GI = retrofit.create(get_item.class);
    }
    return GI;
}

public interface get_item {
    @GET("tracks/{id}?")
    Call<GetTrack> gettrack(@Path("id") String id, @Query("market")
String m);

    @GET("albums/{id}/tracks?")
    Call<GetAlbumTracks> getAlbumTracks(@Path("id") String id);

    @GET("artists/{id}/top-tracks?")
    Call<ArtistTracks> getArtistTracks (@Path("id") String id,
@Query("market") String m);
}
```

TESTING

Software Testing is evaluation of the software against requirements gathered from users and system specifications. Testing is conducted at the phase level in software development life cycle or at module level in program code.

Testing Approaches

Tests can be conducted based on two approaches –

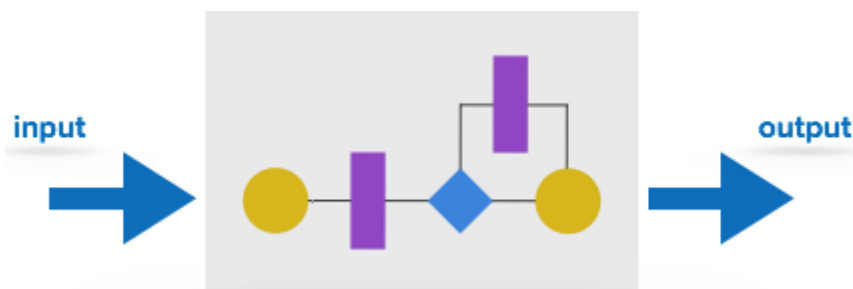
- Functionality testing
- Implementation testing

When functionality is being tested without taking the actual implementation in concern it is known as **black-box testing**. The other side is known as **white-box testing** where not only functionality is tested but the way it is implemented is also analysed.

We have implemented white-box testing in our project.

White-box testing

It is conducted to test program and its implementation, in order to improve code efficiency or structure. It is also known as ‘Structural’ testing.



In this testing method, the design and structure of the code are known to the tester. Programmers of the code conduct this test on the code.

The below are some White-box testing techniques:

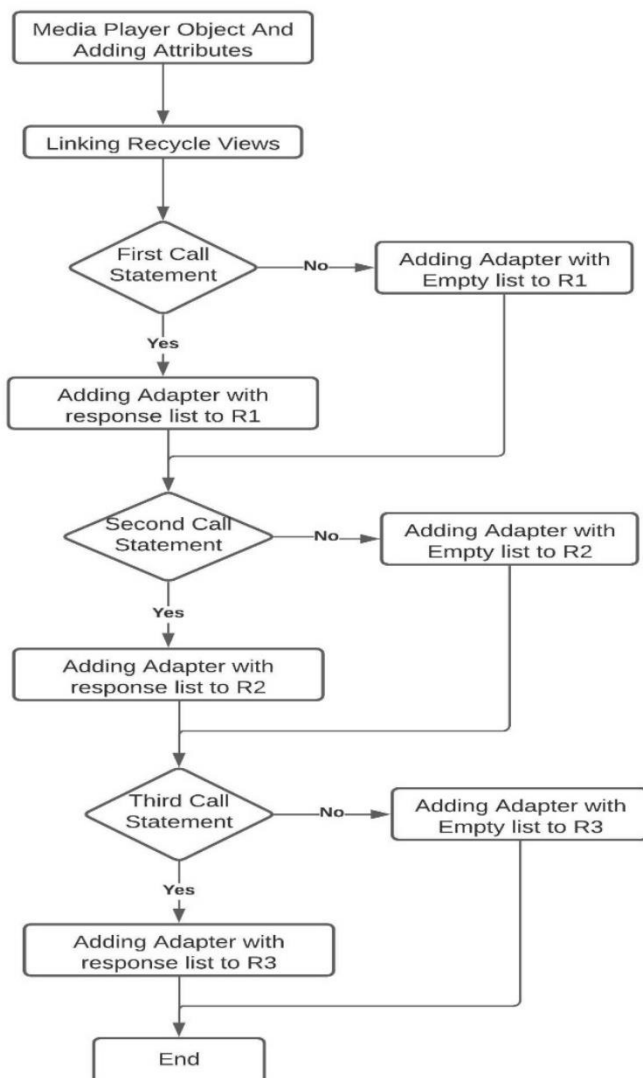
- **Control-flow testing** - The purpose of the control-flow testing to set up test cases which covers all statements and branch conditions. The

branch conditions are tested for both being true and false, so that all statements can be covered.

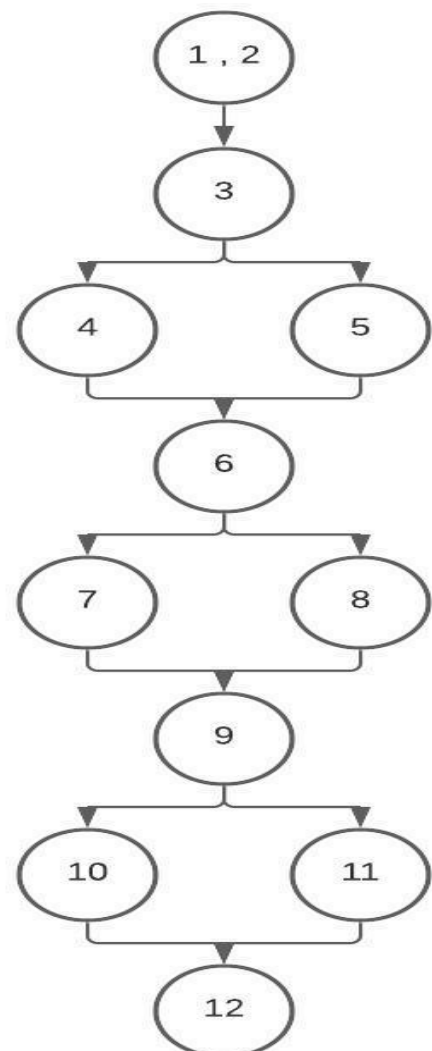
- **Data-flow testing** - This testing technique emphasis to cover all the data variables included in the program. It tests where the variables were declared and defined and where they were used or changed.

FLOWCHART AND FLOWGRAPH FOR OUR PSEUDOCODE

FLOWCHART



FLOWGRAPH



Independent Paths:

1,2,3,4,6,7,9,10,12

1,2,3,4,6,7,9,11,12

1,2,3,4,6,8,9,10,12

1,2,3,4,6,8,9,11,12

1,2,3,5,6,7,9,10,12

1,2,3,5,6,7,9,11,12

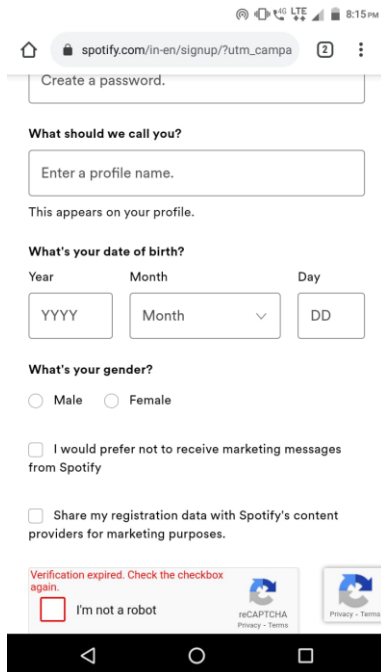
1,2,3,5,6,8,9,10,12

1,2,3,5,6,8,9,11,12

Cyclomatic Complexity = Total no. of independent paths: 8

INTERFACE DESIGN

Sign-Up



spotify.com/in-en/signup/?utm_campa

Create a password.

What should we call you?

Enter a profile name.

This appears on your profile.

What's your date of birth?

Year Month Day

YYYY Month DD

What's your gender?

☐ Male ☐ Female

☐ I would prefer not to receive marketing messages from Spotify

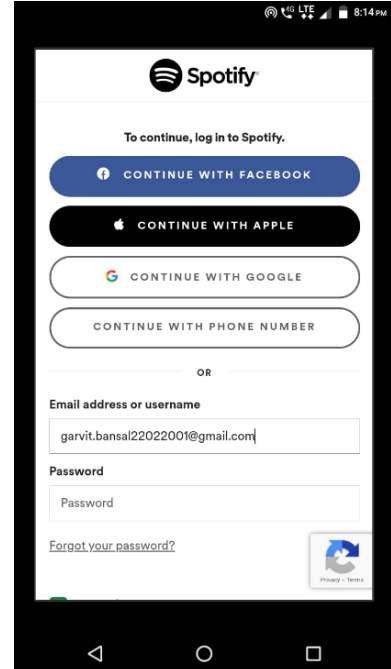
☐ Share my registration data with Spotify's content providers for marketing purposes.

Verification expired. Check the checkbox again.

☐ I'm not a robot

reCAPTCHA Privacy - Terms

Login



Spotify

To continue, log in to Spotify.

[CONTINUE WITH FACEBOOK](#)

[CONTINUE WITH APPLE](#)

[CONTINUE WITH GOOGLE](#)

[CONTINUE WITH PHONE NUMBER](#)

OR

Email address or username

garvit.bansal2022001@gmail.com

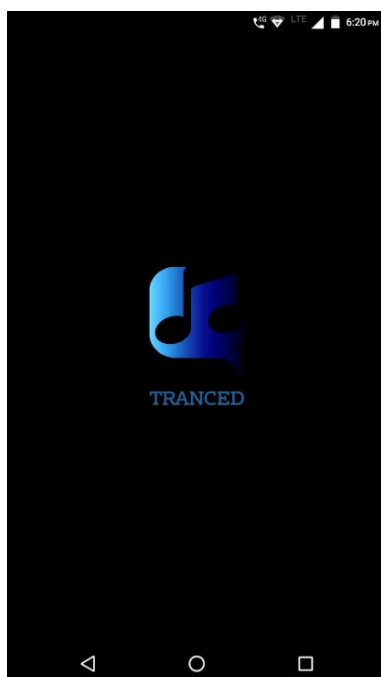
Password

Password

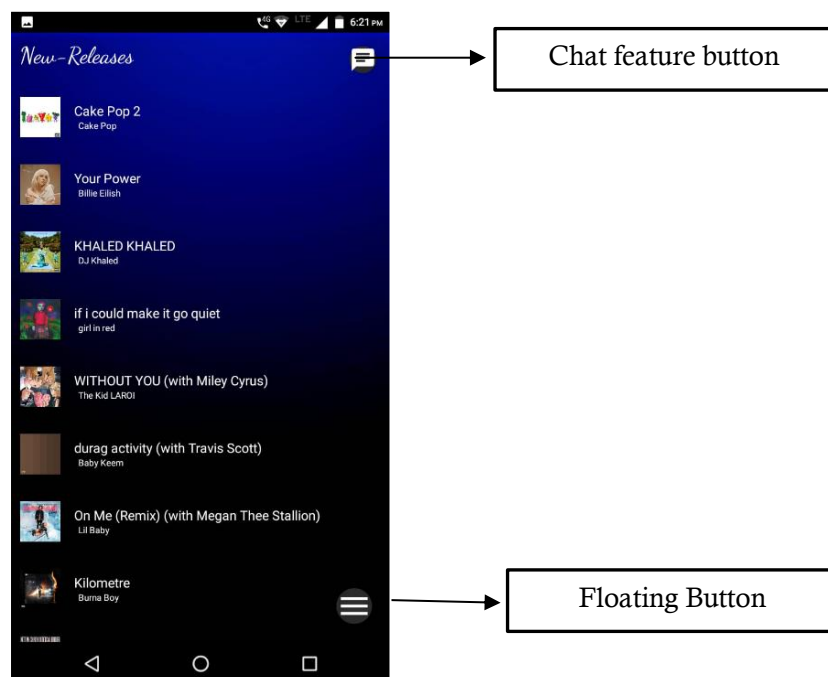
[Forgot your password?](#)

Privacy - Terms

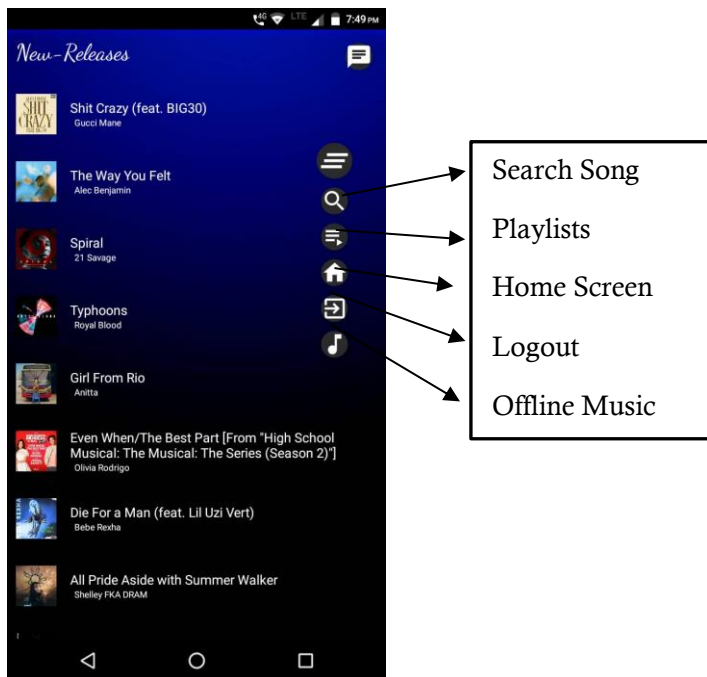
Splash Screen



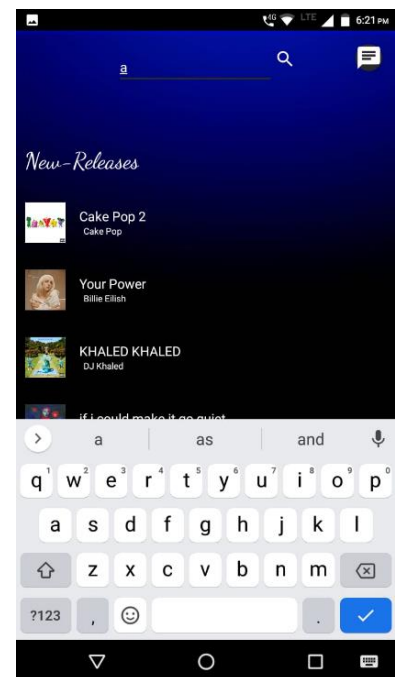
New Releases on Home Screen



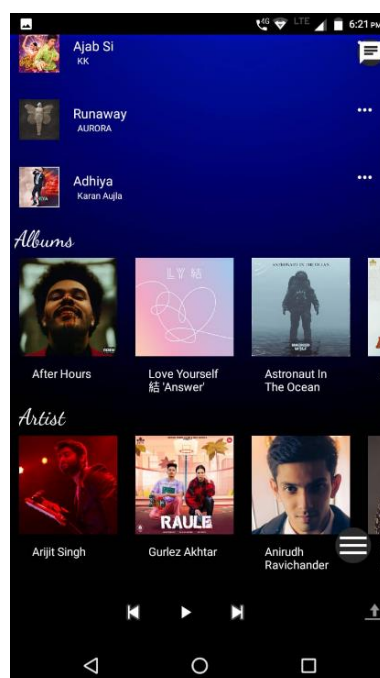
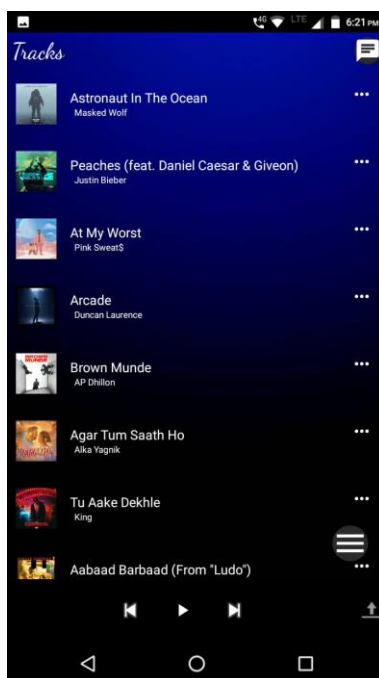
Floating Button Options



Search Song



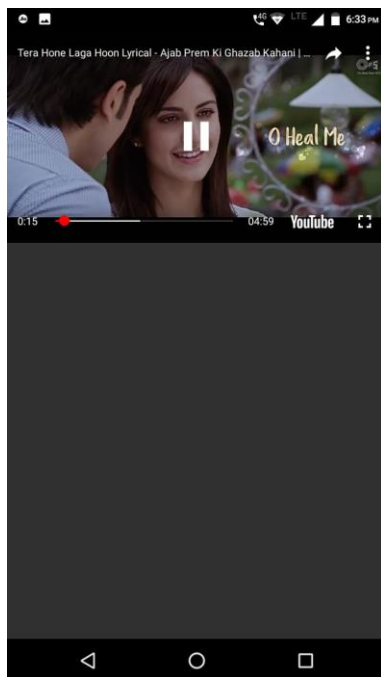
Search Song Result: Tracks, Albums, Artists



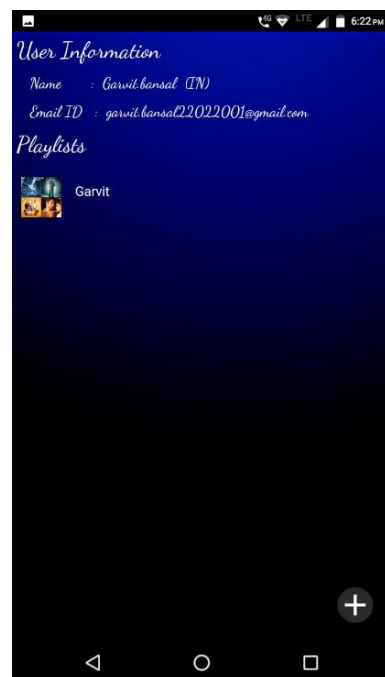
Music Player Options:

- Play
- Pause
- Previous
- Next
- YouTube Window

YouTube Window: Song Video

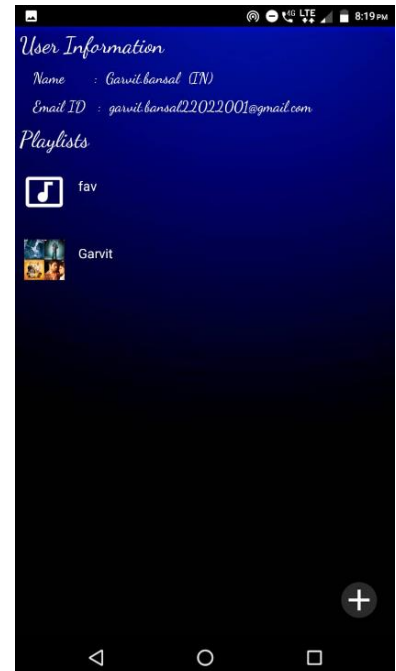
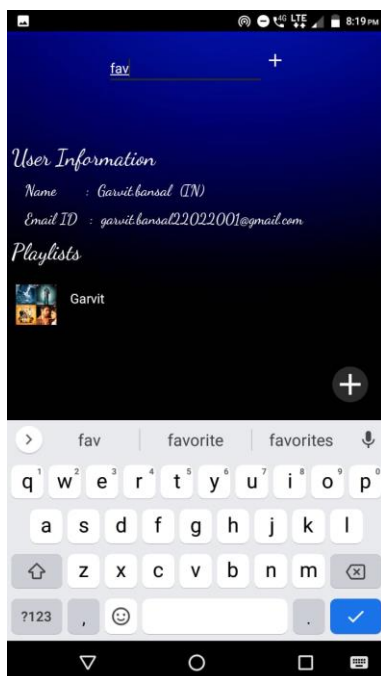


Playlist



Create playlist

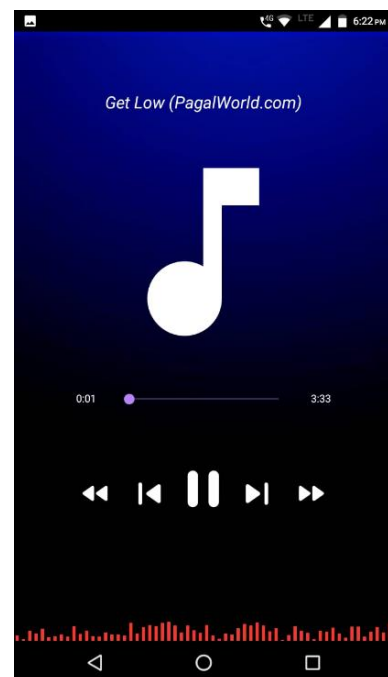
New Playlist Created



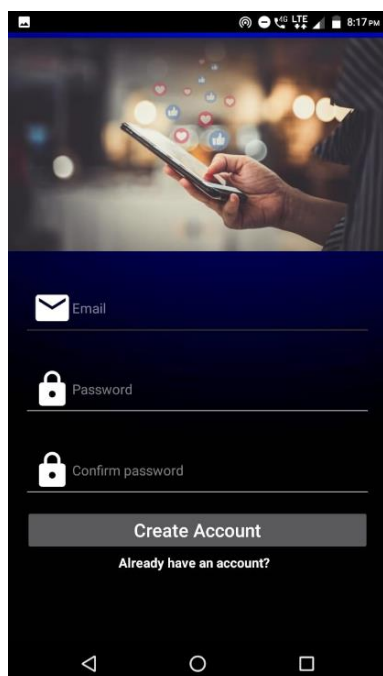
Offline Music



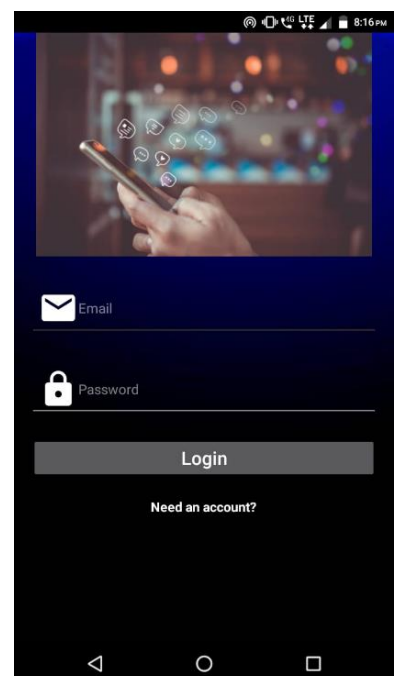
Music Player for offline music



Sign-Up for Chat Feature

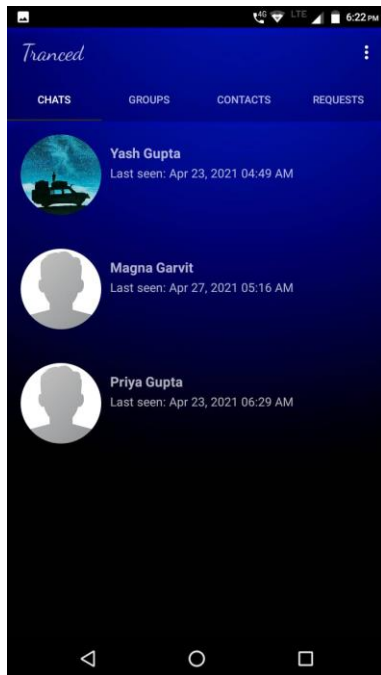


Login for Chat Feature



Home Screen: Chats, Groups, Contacts, Requests

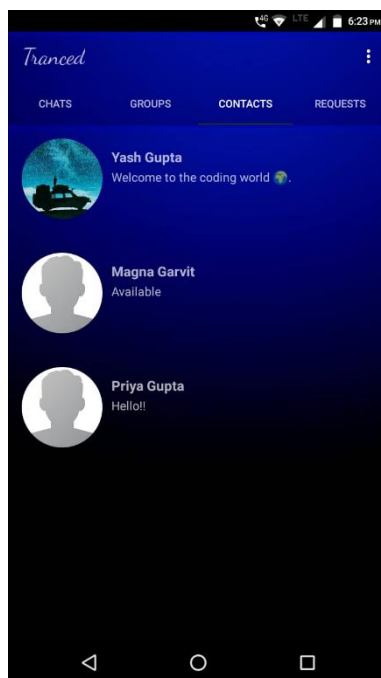
Chats



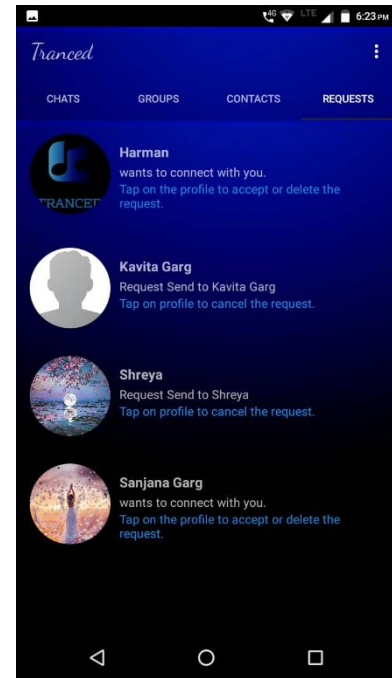
Groups



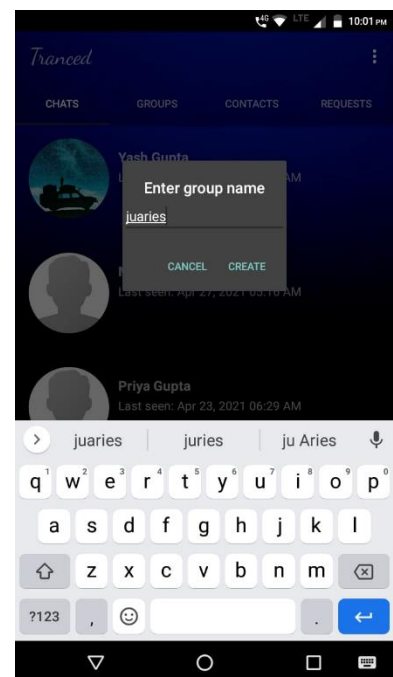
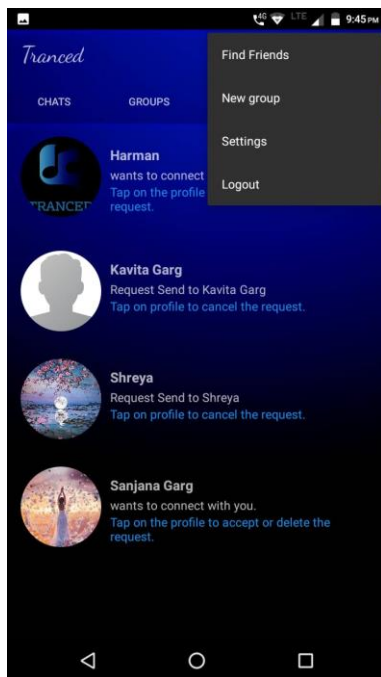
Contacts



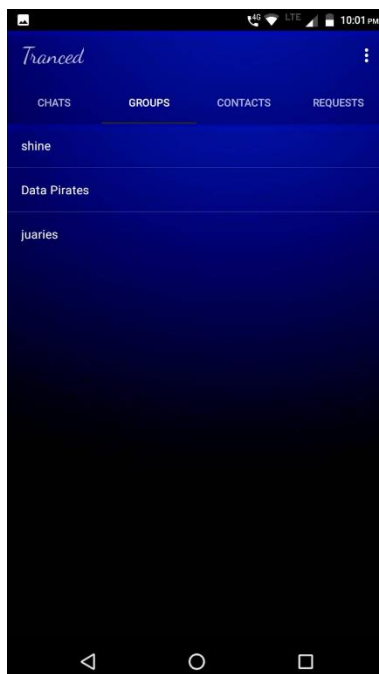
Requests



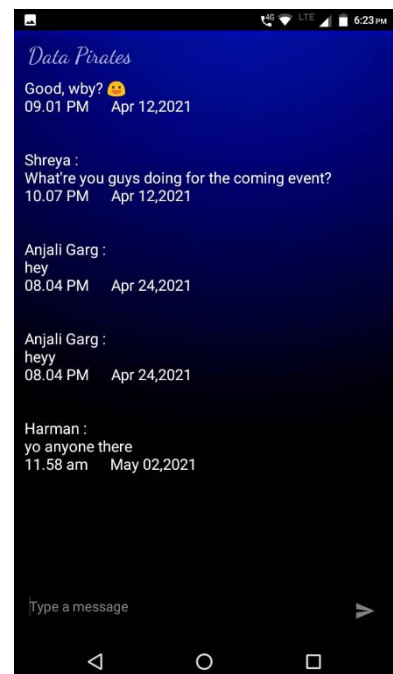
Create New Group



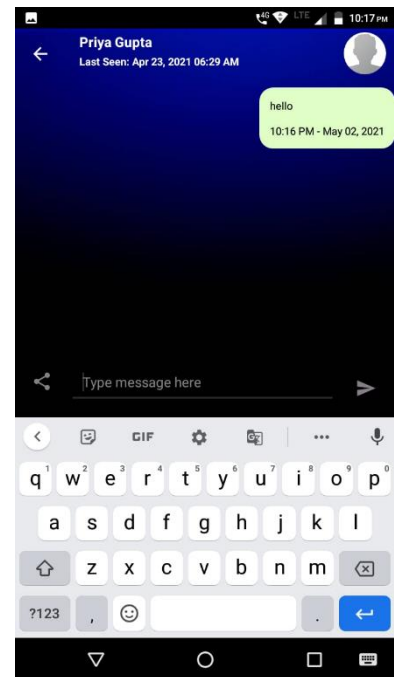
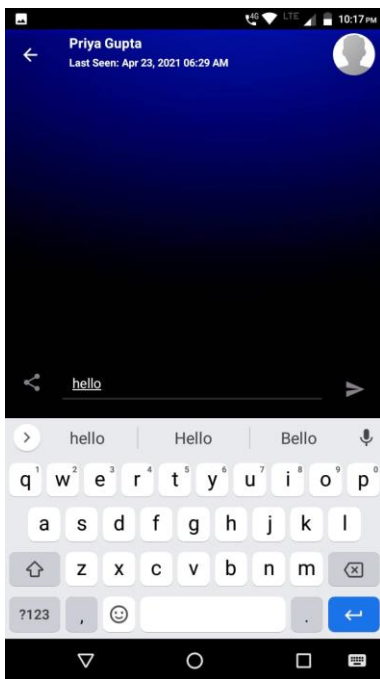
Created Groups



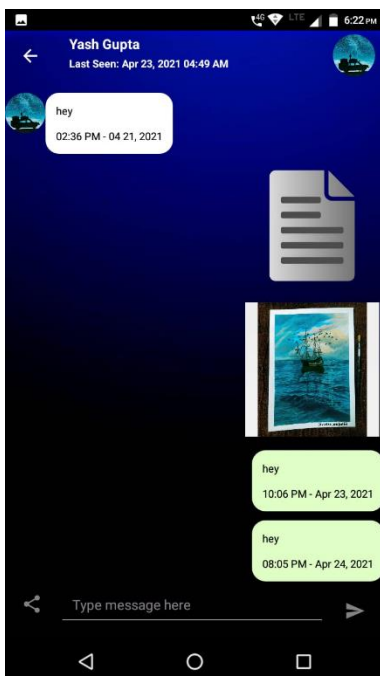
Group Chat



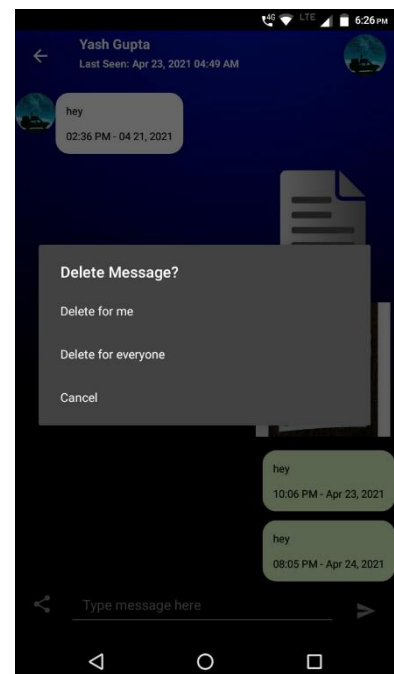
Personal Chat



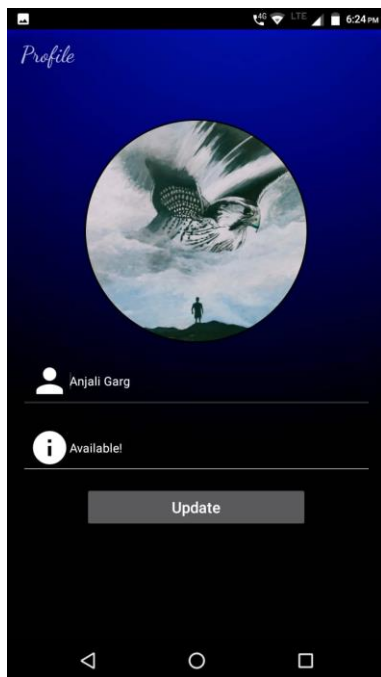
Images and Documents in chat



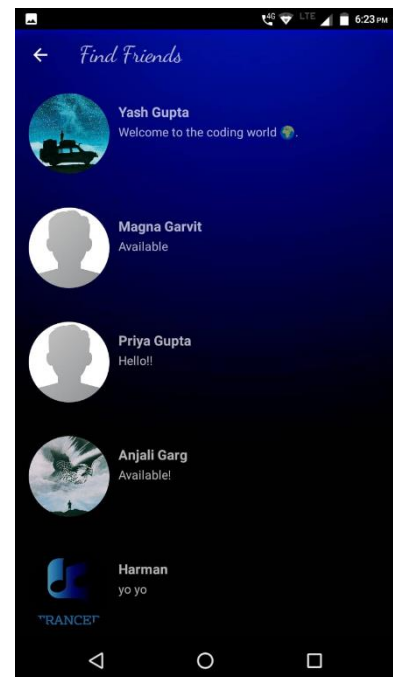
Delete message options



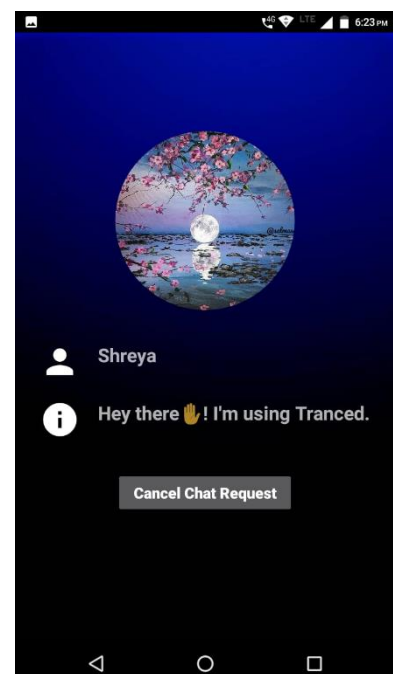
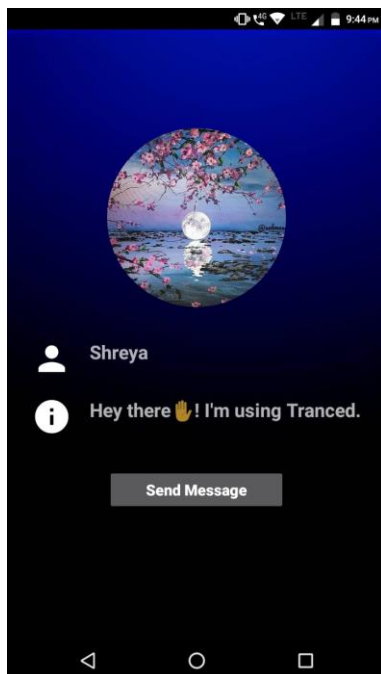
Settings



Find Friends: All App Users



Sending Request



Accepting or deleting the request

