



# MUSIC HOST INTERFACE

**Student:** Thomas Flynn

**Supervisor:** Brian O' Shea

# CONTENTS

- Background
- Tools
- JavaFX Overview
- Azure SQL database
- JavaFX Music Host Application
- Bluetooth
- Android Application
- Time Log
- Conclusion



# BACKGROUND

- Personal
  - Demonstrate Abilities
  - Realization of Ideas
- Initial Idea
  - Virtual Jukebox
- Complete Idea
  - Music Host dictates what the guests can do.
  - The guest can...
    - Request songs
    - Skip songs
    - Give the DJ feedback



# TOOLS

## ○ JavaFX

- Scene Builder
- Media API



## ○ Microsoft Azure Cloud Storage

- User Login Credentials
- User Songs

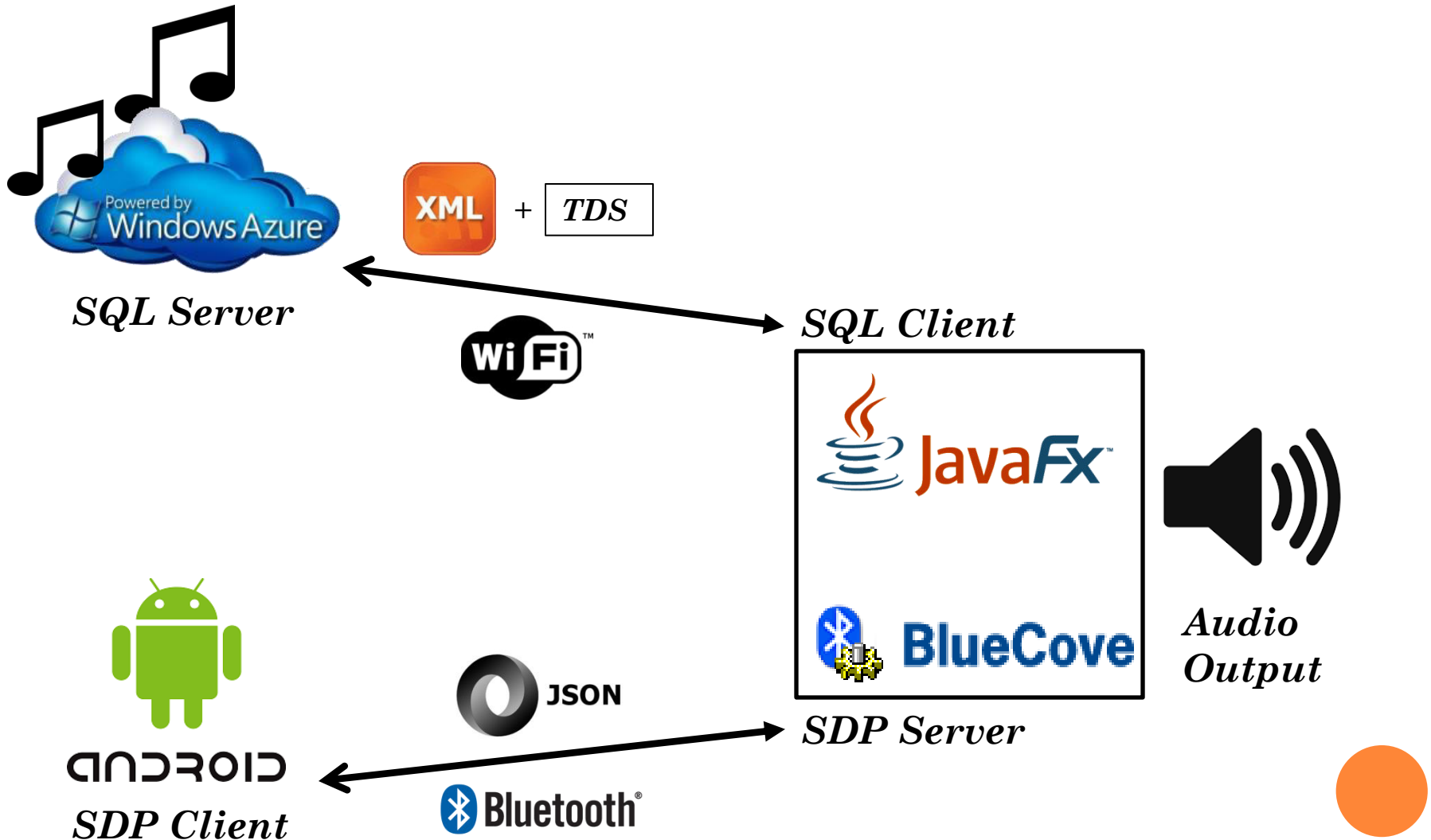


## ○ Bluetooth

- BlueCove library
- Android Bluetooth API



# SYSTEM BLOCK DIAGRAM



# CHRISTMAS PROJECT POSTER

## Music Host Interface



Thomas Flynn

BACHELOR OF ENGINEERING IN COMPUTER AND ELECTRONIC ENGINEERING

A Music Player Application that interacts with both an Android App and a music database on the cloud.

"The solution to music selection"



- Can access music from anywhere
- Lets users store and delete songs
- Outputs songs to music player
- User's songs stored separately
- Deals with login accounts

### Primary features

- Play music files from cloud storage
- Skip songs
- Rearrange songs in queue
- View all available songs on the cloud
- Login screen
- Primary music player screen
- Disk Jockey Screen
- Disk Jockey can take song requests



JavaFx

Bluetooth ANDROID



- Request to view the database
- Enables Bluetooth
- Queue a song from database
- User can view the song queue
- Easy to use
- Save song files from database
- Transmit song files to host

### Secondary features

- Next song fades in as the other fades out
- DSP filters for mixing audio
- Queue reorders depending on votes
- Automated responses for Android client
- Switch to turntable screen for DJ

Song Request Life cycle



### JavaFX Scene switching

#### Login Scene



#### Media Player Scene



#### Disk Jockey Scene



# JAVAFX OVERVIEW

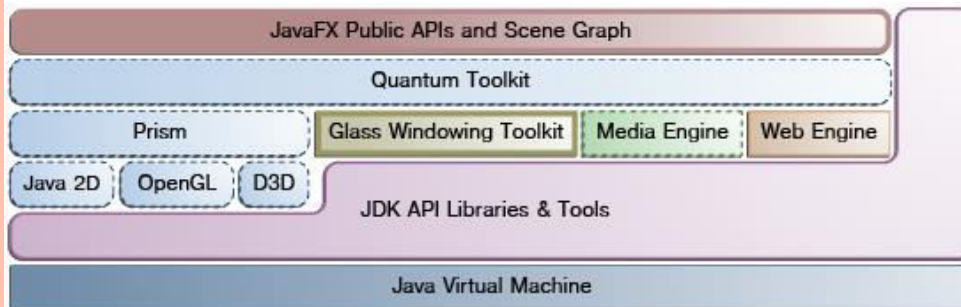
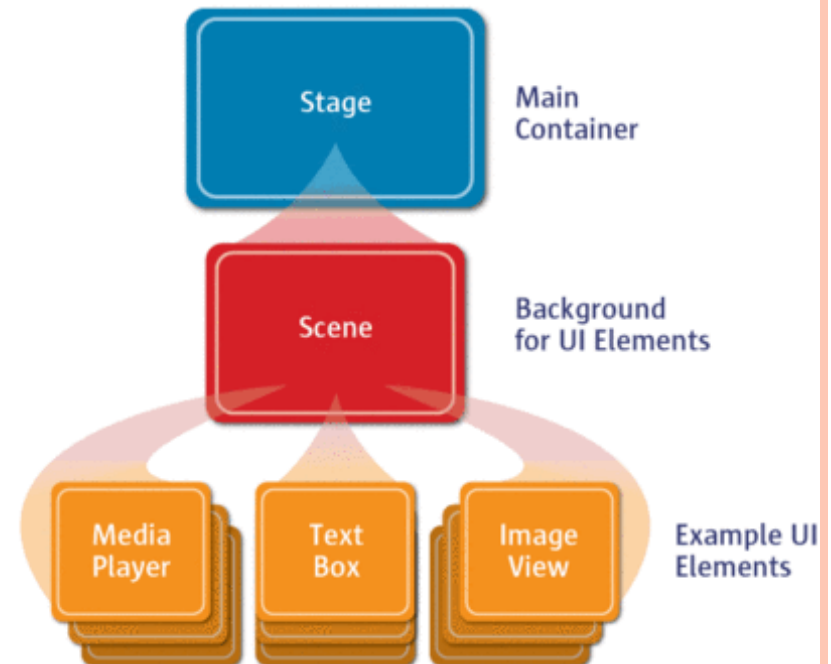
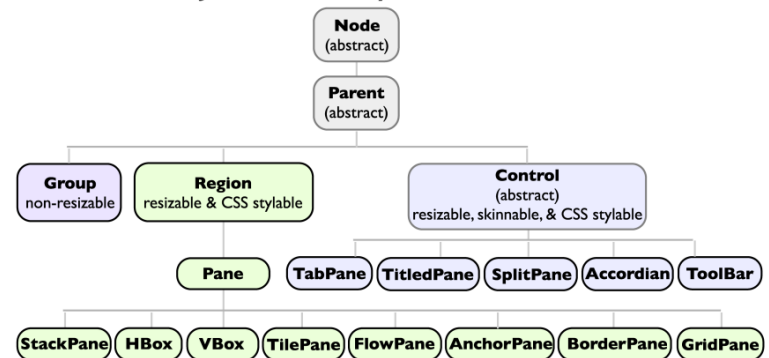
- **JavaFX** is intended to replace Swing as the standard GUI library for Java SE.
- **Swing interoperability**- Existing Swing applications can be updated with new JavaFX features.
- **FXML** is an XML-based declarative markup language for constructing a JavaFX application user interface.
- **Scene Builder** generates FXML markup that can be ported to an IDE where a developer can add the business logic.



# JAVAFX OVERVIEW

- Stage.
- Scene Graph.
- Node.
- Includes graphics primitives.
- JavaFX application thread.
- Media thread.

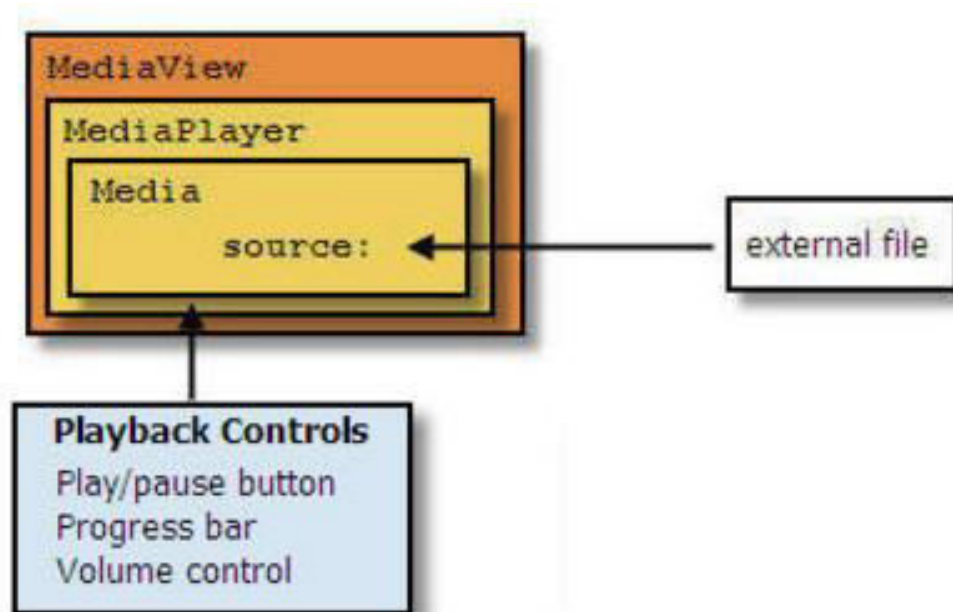
JavaFX 2.0 Layout Classes





# MEDIA IN JAVA FX

- Media classes are part of javafx.scene.media package.
- Media**, **MediaPlayer** and **MediaView**
  - MediaView** is a Node, it has a **MediaPlayer**.
  - MediaPlayer** has a **Media** object.
- MediaTimers allow functions to be invoked at key points in Media.





# MICROSOFT AZURE SQL DATABASE

- Cloud based service (DBaaS)
- Special version of Microsoft SQL Server as its backend.
- XML based format for data transfer.
- T-SQL as the query language.
- Tabular Data Stream (TDS) as the protocol to access the service over the Internet.





# MICROSOFT AZURE SQL DATABASE

- **Minimum Driver Version:** Microsoft JDBC Driver 4.0
- **Driver =**  
`com.microsoft.sqlserver.jdbc.SQLServerDriver`
- **Server=**  
`jdbc:sqlserver://muzikhostserver.database.windows.net:1433;"`
- **Database =** FYP
- **User =** thomas11811@muzikhostserver
- **Password =** Zqlllx\$8





# MICROSOFT AZURE SQL DATABASE

Microsoft Azure

Check out the new portal

Subscriptions

G00291875@gmit.ie

SQL DATABASES  
1

sql databases

DATABASES SERVERS DELETED DATABASES

NAME	STATUS	REPLICATION	LOCATION	SUBSCRIPTION	SERVER	EDITION	MAX SIZE
FYP	Online	None	West Europe	Pay-As-You-Go	muzikhostserver	Basic	2 GB

UserLogin
Id int
userName varchar(255)
password varchar(255)

Id:Id

Database

Server

muzikhostserver.database.windows.net

Databases

System Databases

FYP

Tables

System Tables

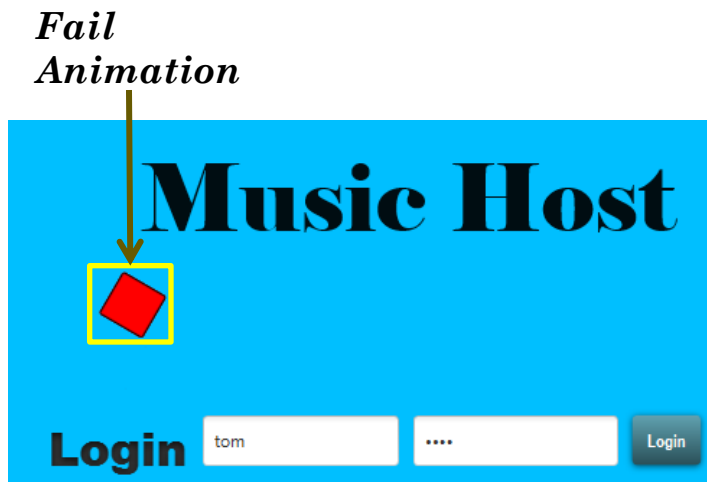
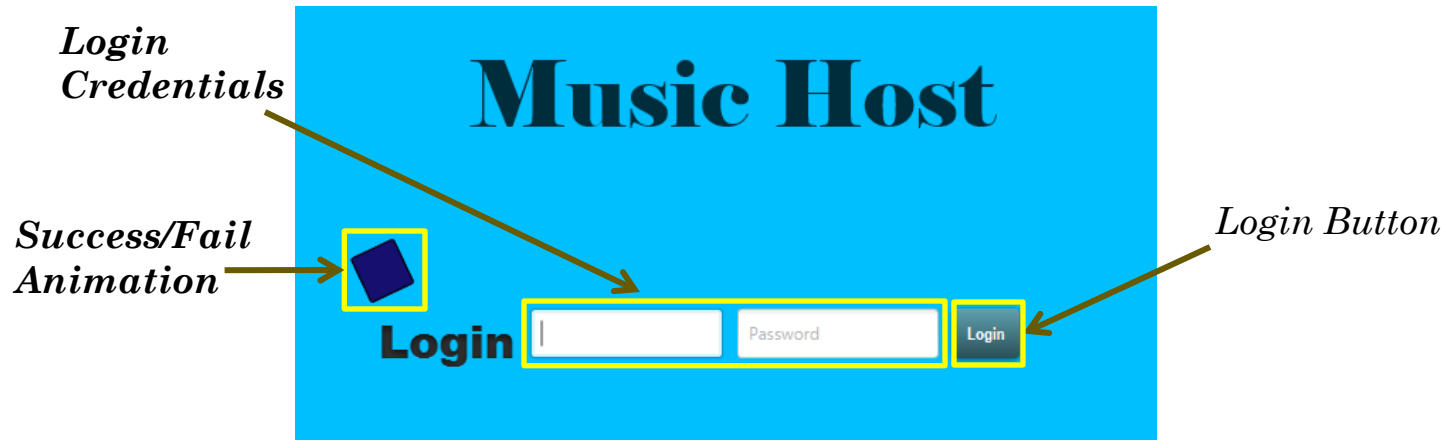
dbo.Persons

dbo.UserLogin

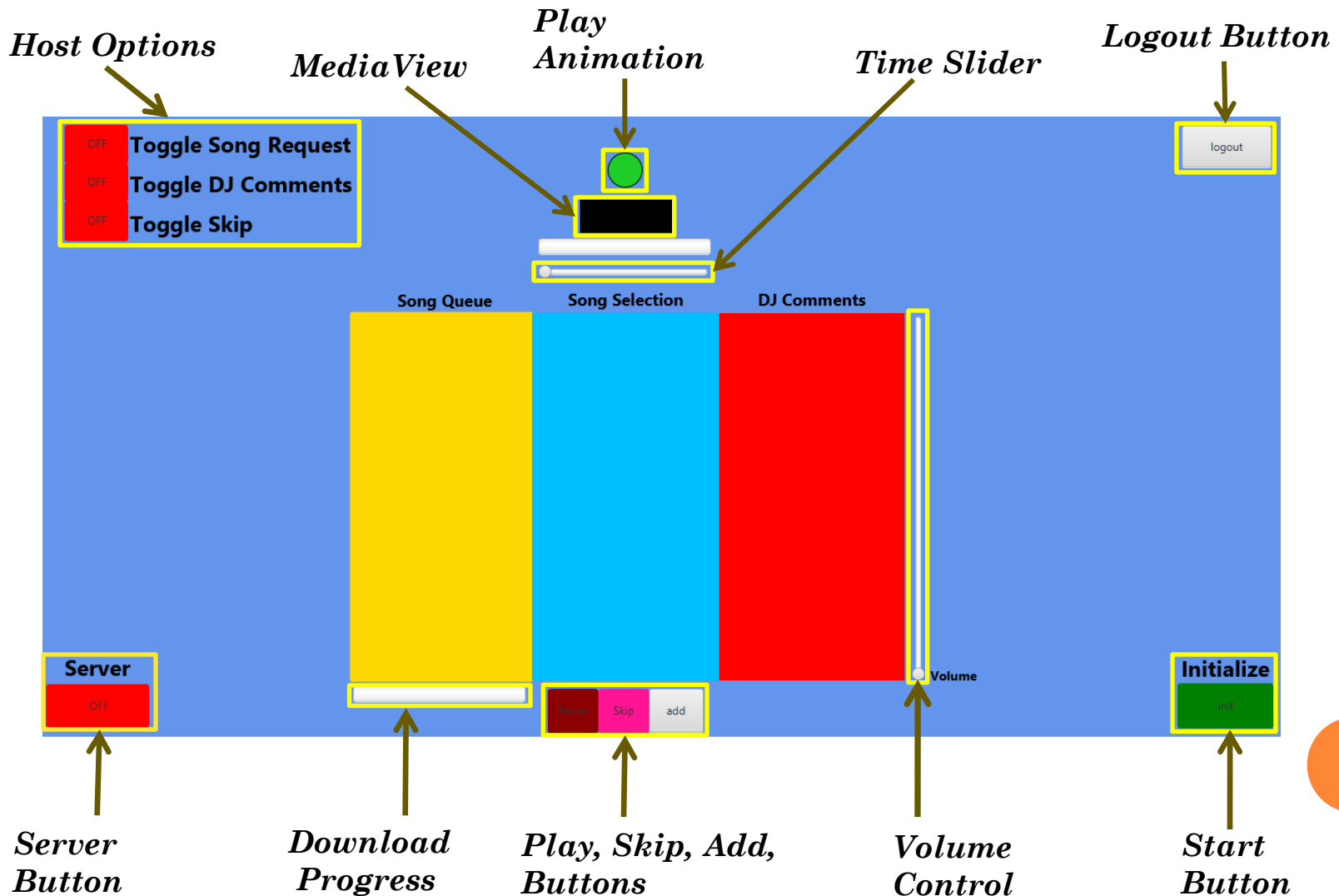
dbo.UserSongs

UserSongs
S_Id int
songName varchar(255)
artistName varchar(255)
dataSize int
data varbinary(max)
Id int

# MUSIC HOST - LOGIN SCREEN



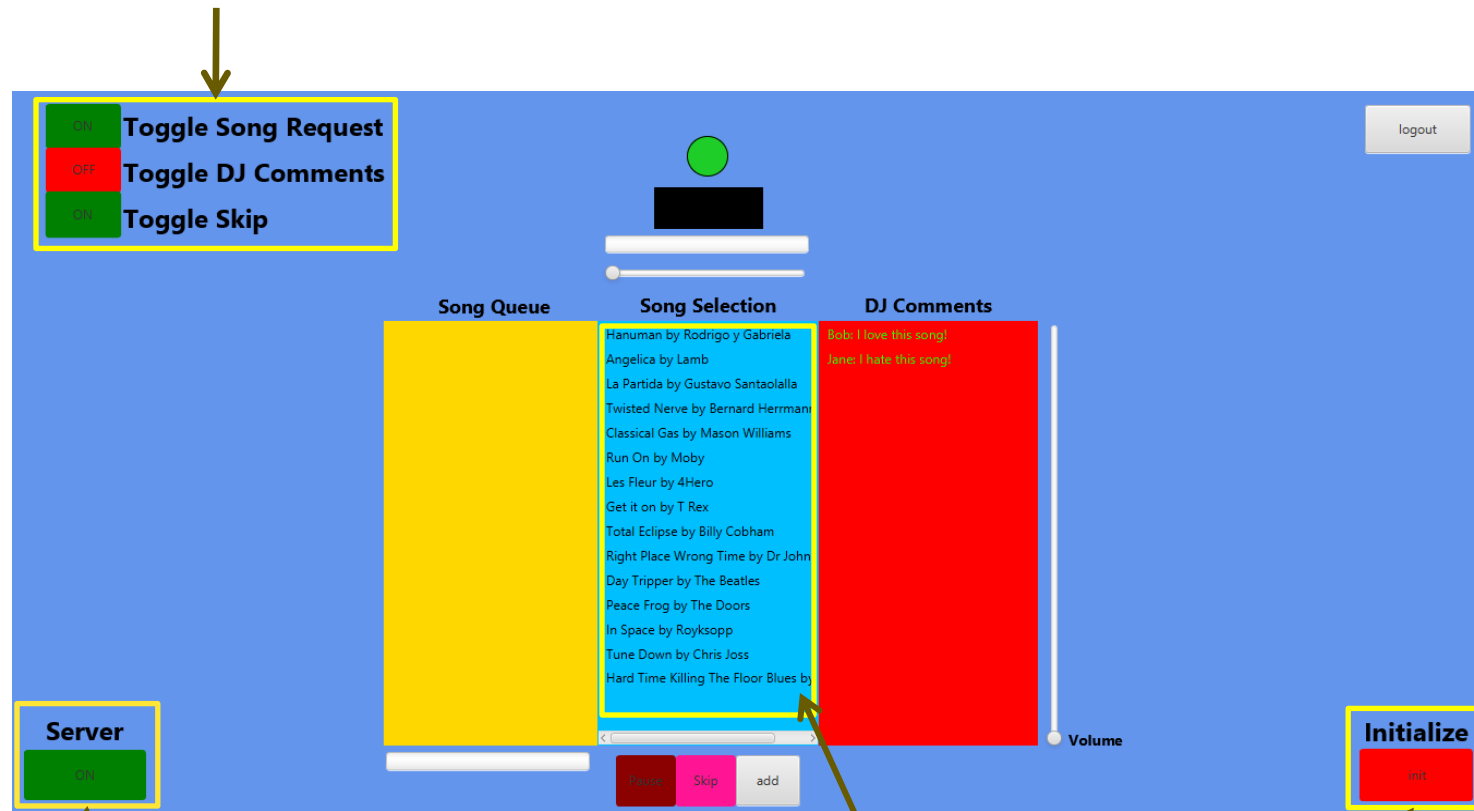
# MUSIC HOST - MAIN SCREEN



# MUSIC HOST

## USE CASE – SETUP

*Guest Options Specified*



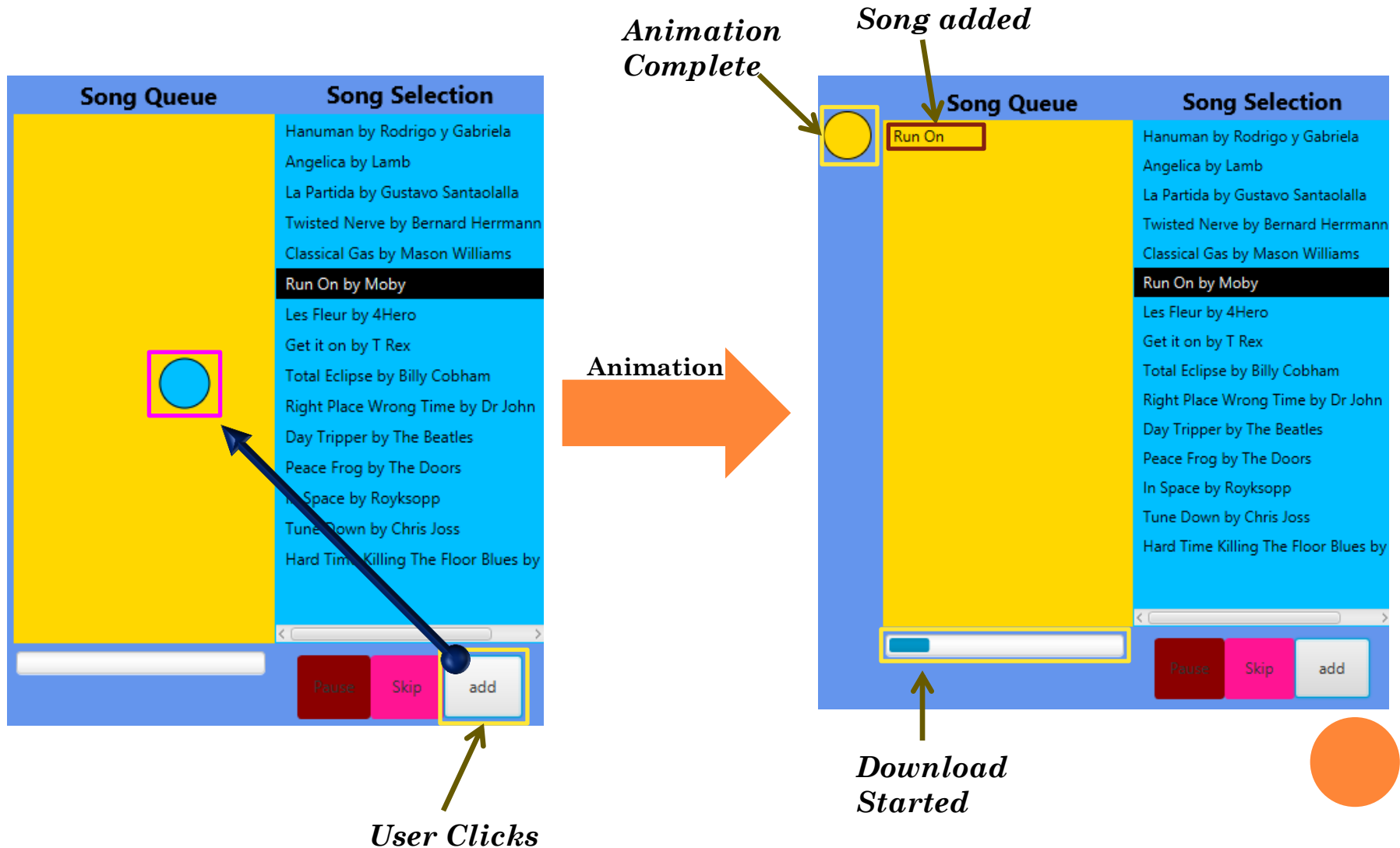
*Server Enabled*

*Selection Populated*

*Initialized*

# MUSIC HOST

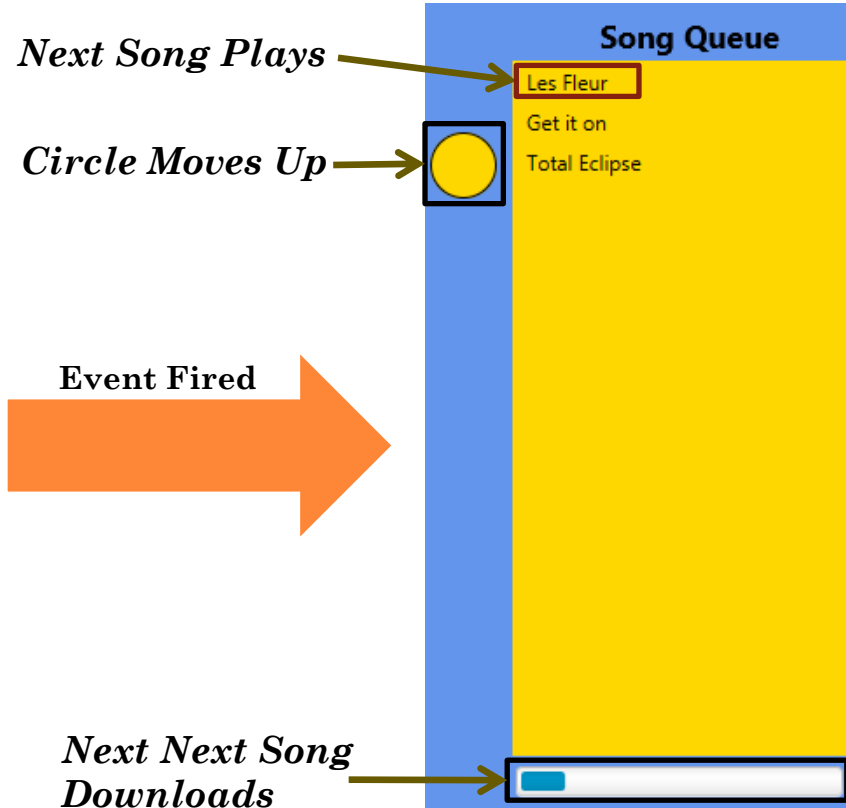
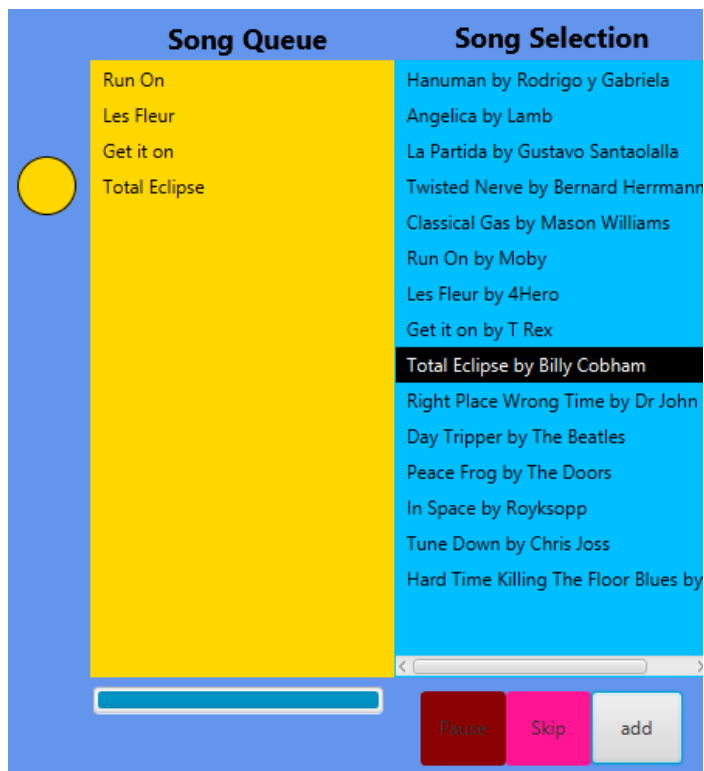
## USE CASE - SONG ADDED



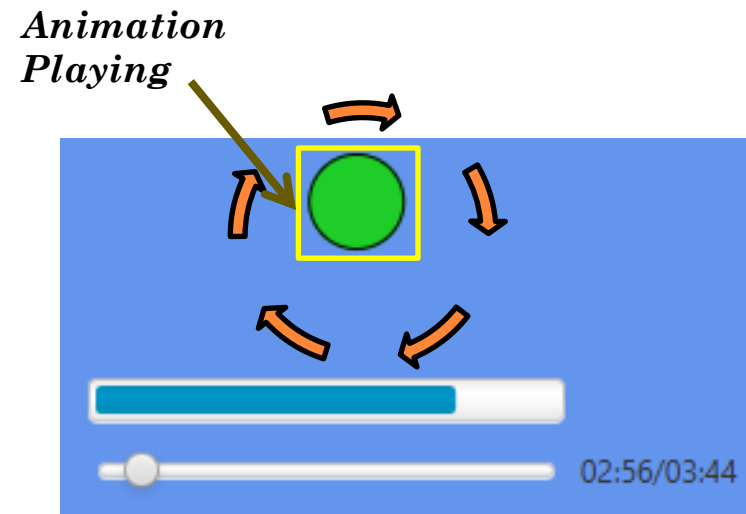
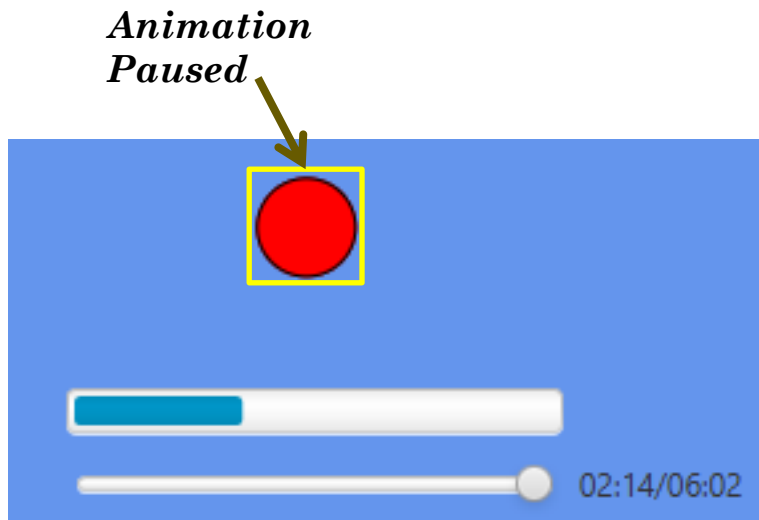


# MUSIC HOST

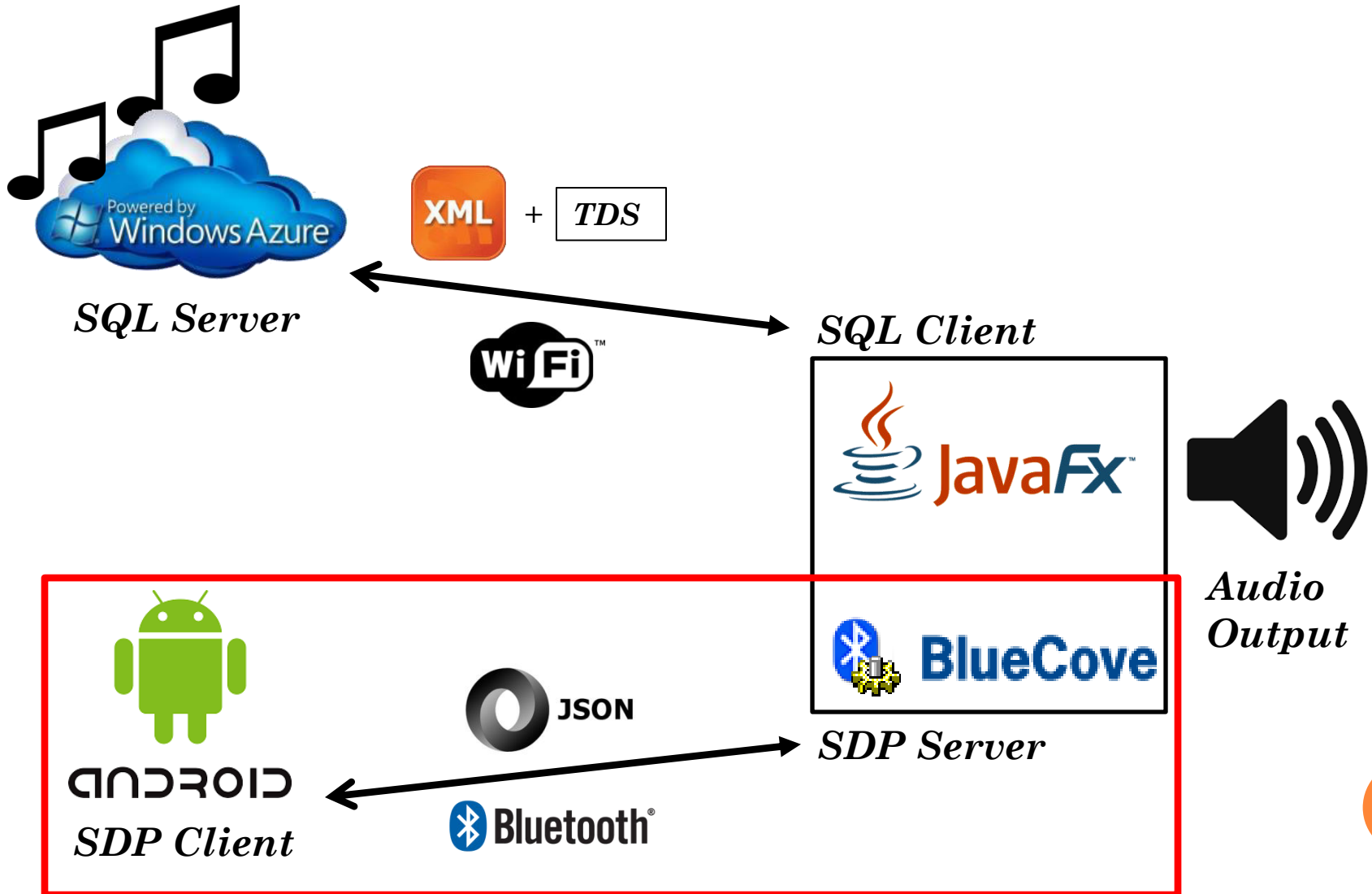
## USE CASE - SONG SKIPPED / SONG ENDED



# MUSIC HOST USE CASE – PLAY / PAUSE



# SYSTEM BLOCK DIAGRAM





## BLUETOOTH SERVER

- **Profile:** Serial Port Profile
- One connection at a time.
- Service discovery procedure has to be performed to set up an emulated serial cable connection.
- Devices are paired during the connection establishment phase.
- Link establishment is initiated by Android Client.
- There are no fixed master slave roles.
- RFCOMM is used to transport the user data, modem control signals and configuration commands.



# SERIAL PORT PROFILE CONNECTION



ANDROID

***SDP Client***

**1.1** Submit a query using SDP.

**2.1** Request a new L2CAP channel.

**3.1** Initiate an RFCOMM session.

**4.1** Start a new data link connection on the RFCOMM session.



**BlueCove**

***SDP Server***

**1.2** Authentication procedure.

**2.2** Accept a new L2CAP channel.

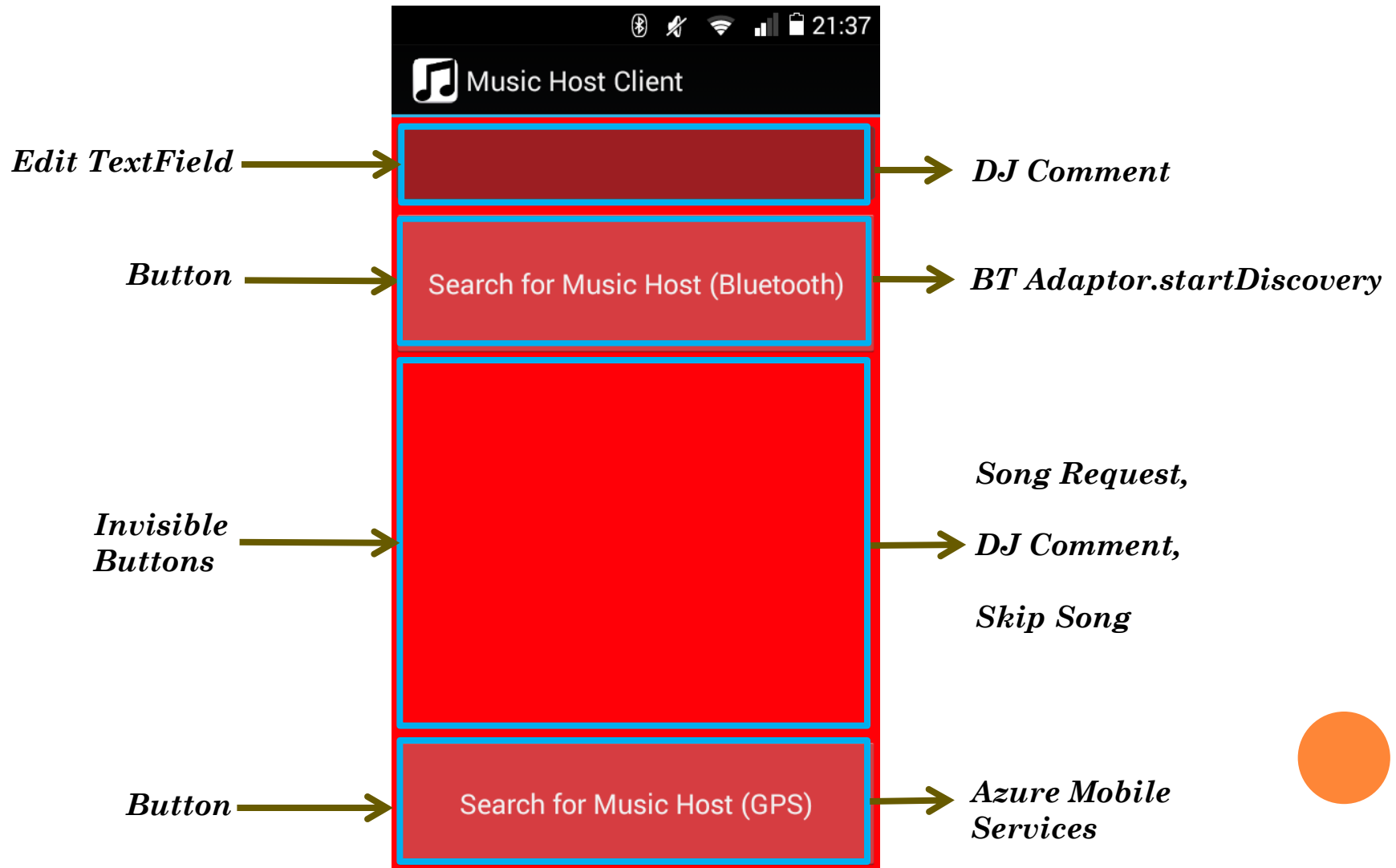
**3.2** Accept an RFCOMM session establishment on that channel.

**4.2** Accept a new data link connection on the RFCOMM session.



# MUSIC HOST CLIENT

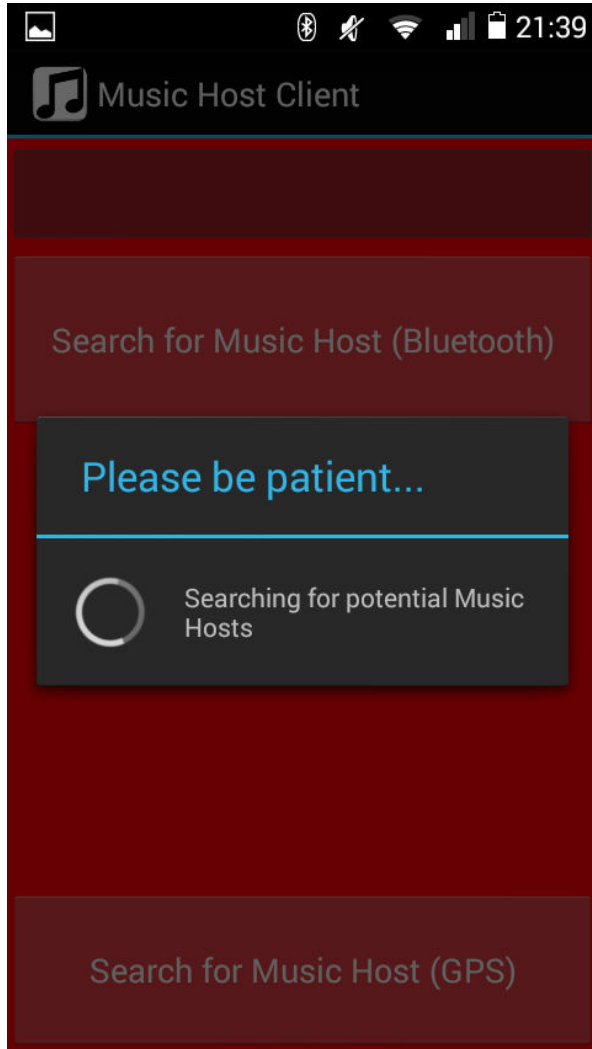
## USE CASE – OPEN APP



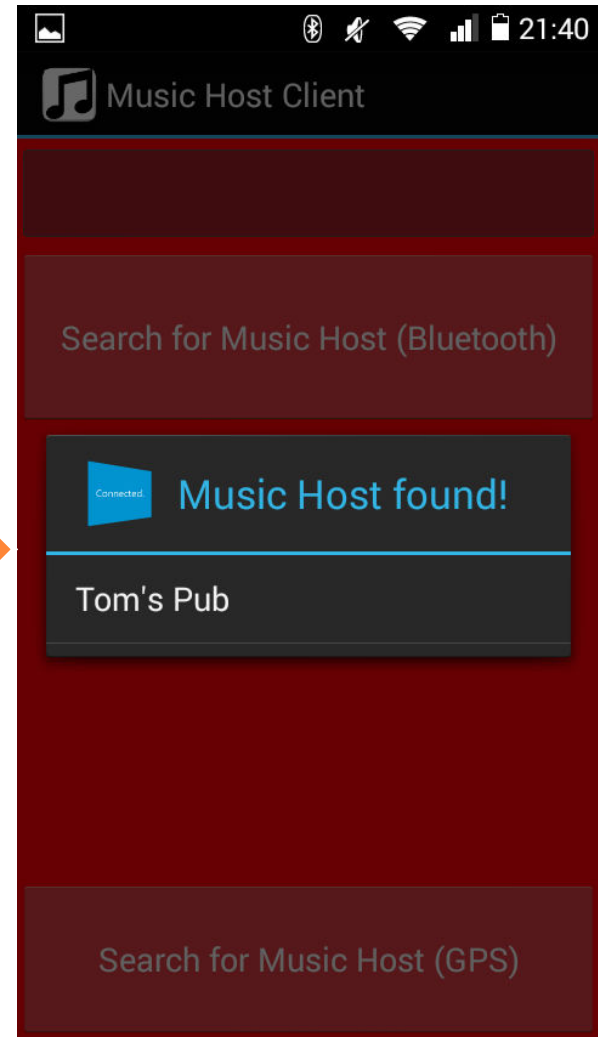
# MUSIC HOST CLIENT

## USE CASE –

### *SEARCH FOR MUSIC HOST (BT)*

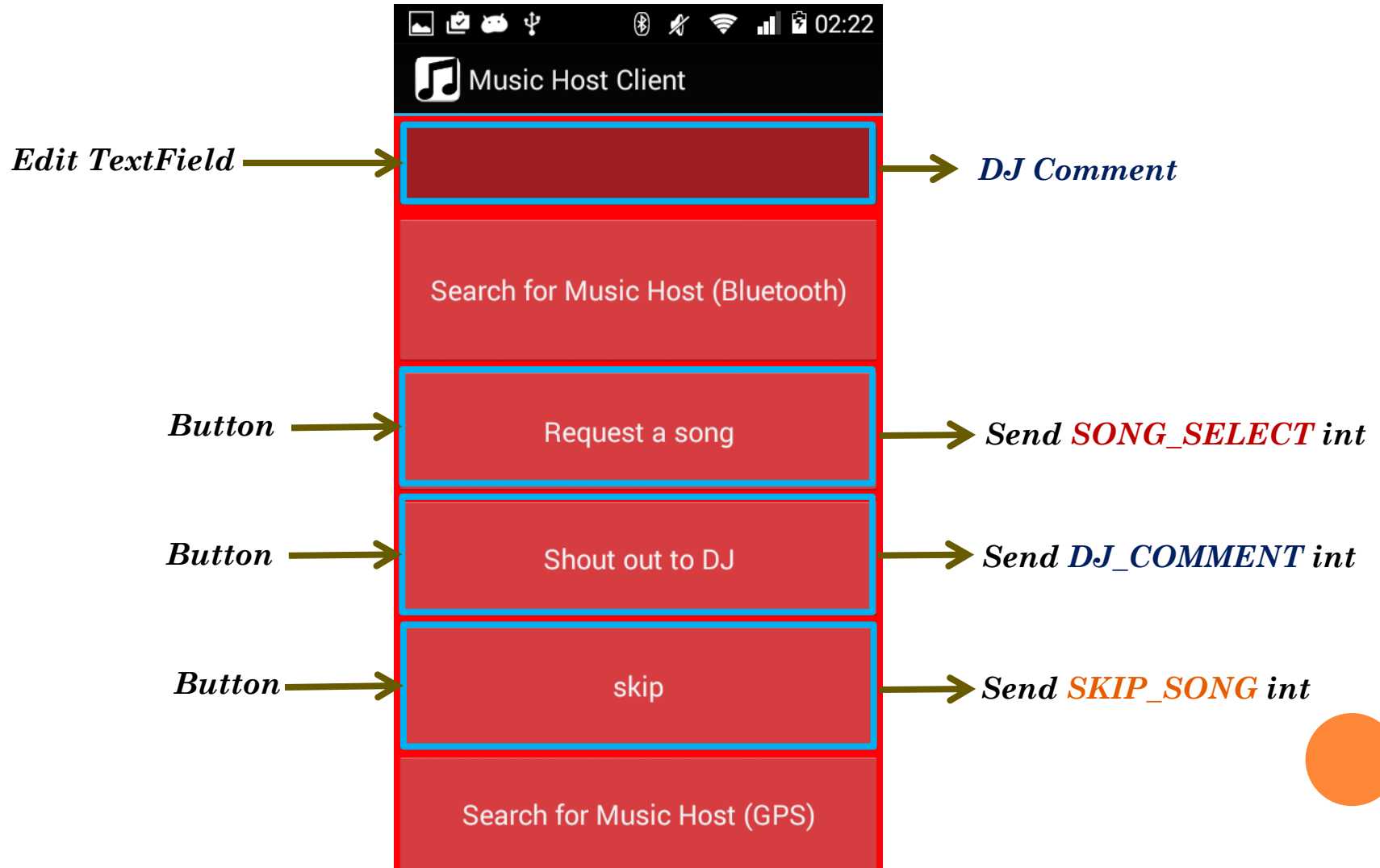


*Music Host Found*



# MUSIC HOST CLIENT

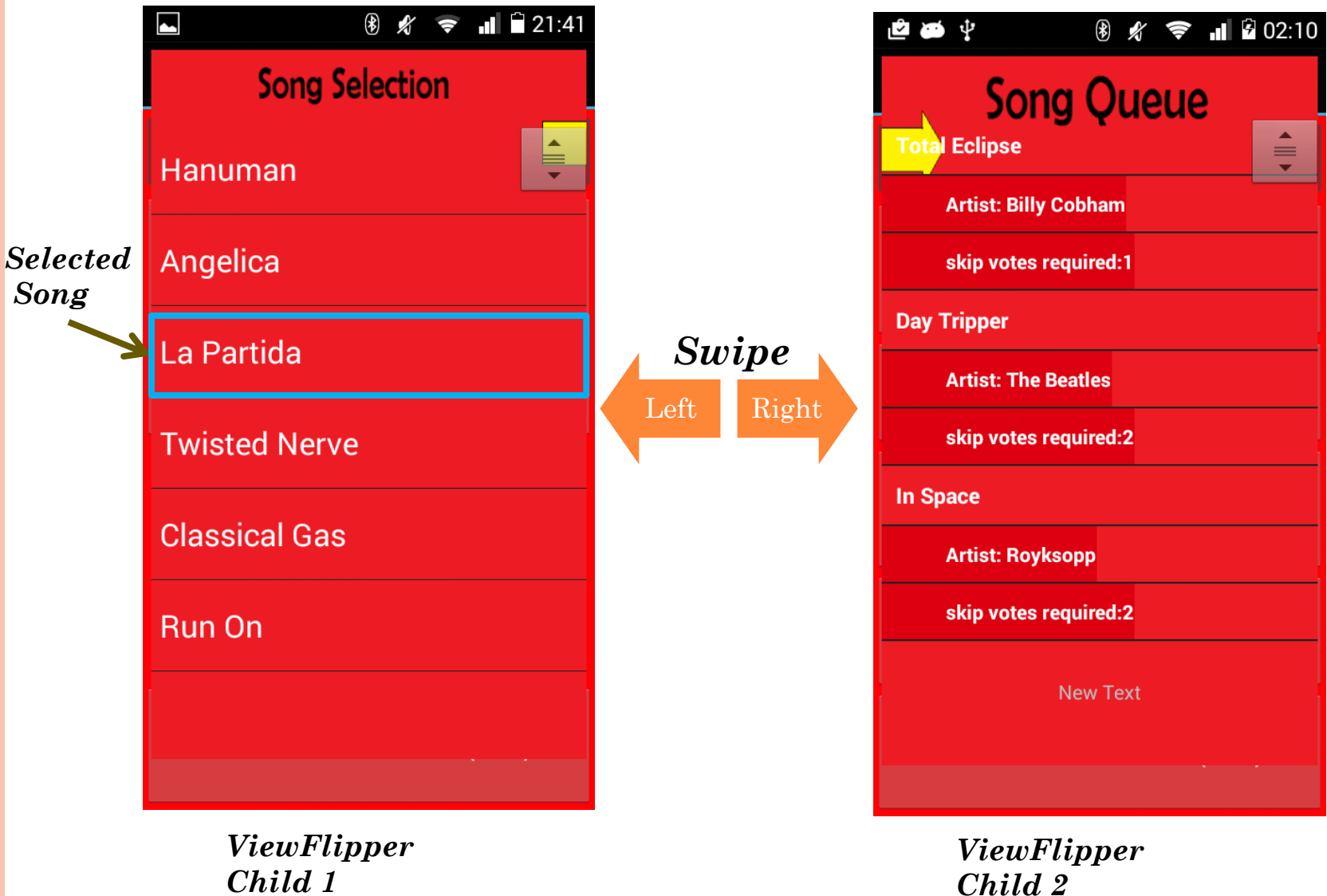
## USE CASE – CONNECTED





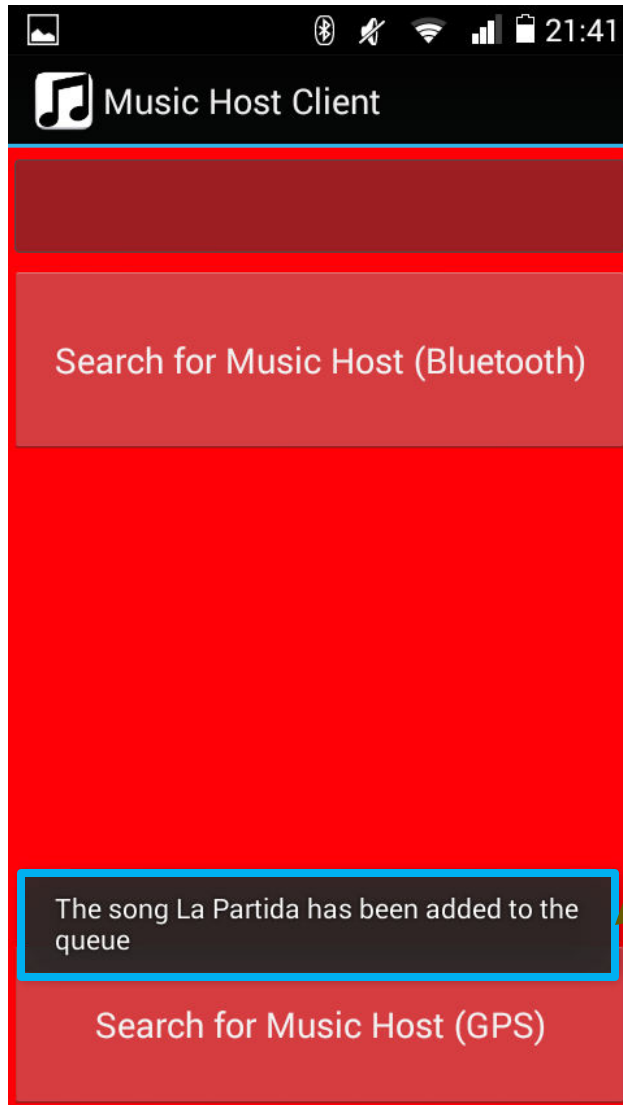
# MUSIC HOST CLIENT

## USE CASE – SONG REQUEST

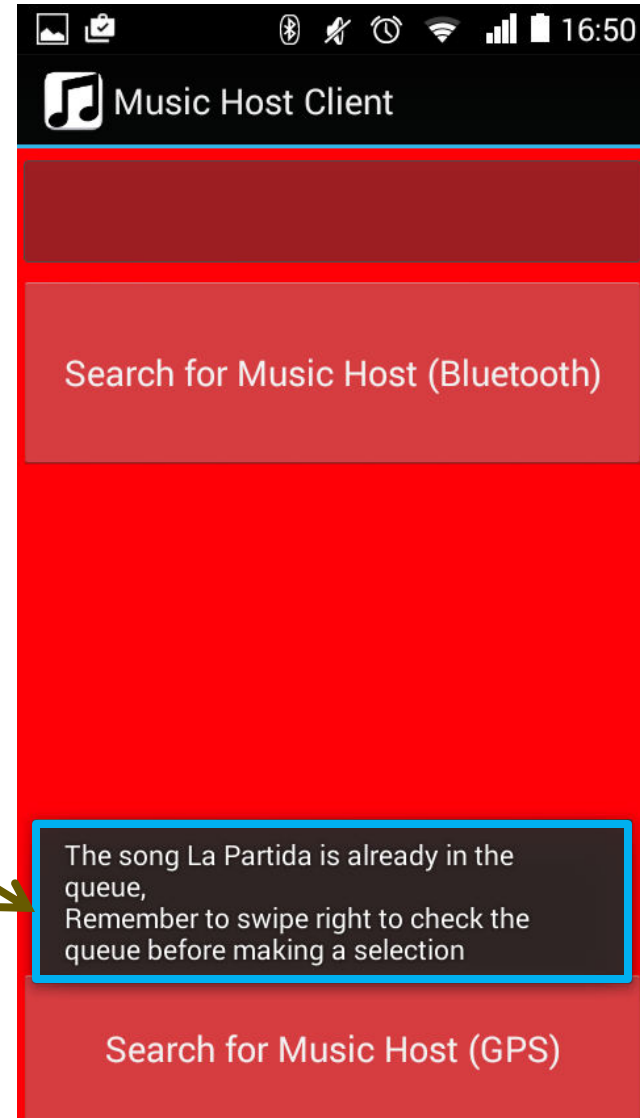


# MUSIC HOST CLIENT

## USE CASE – SONG ACCEPTED / NOT ACCEPTED

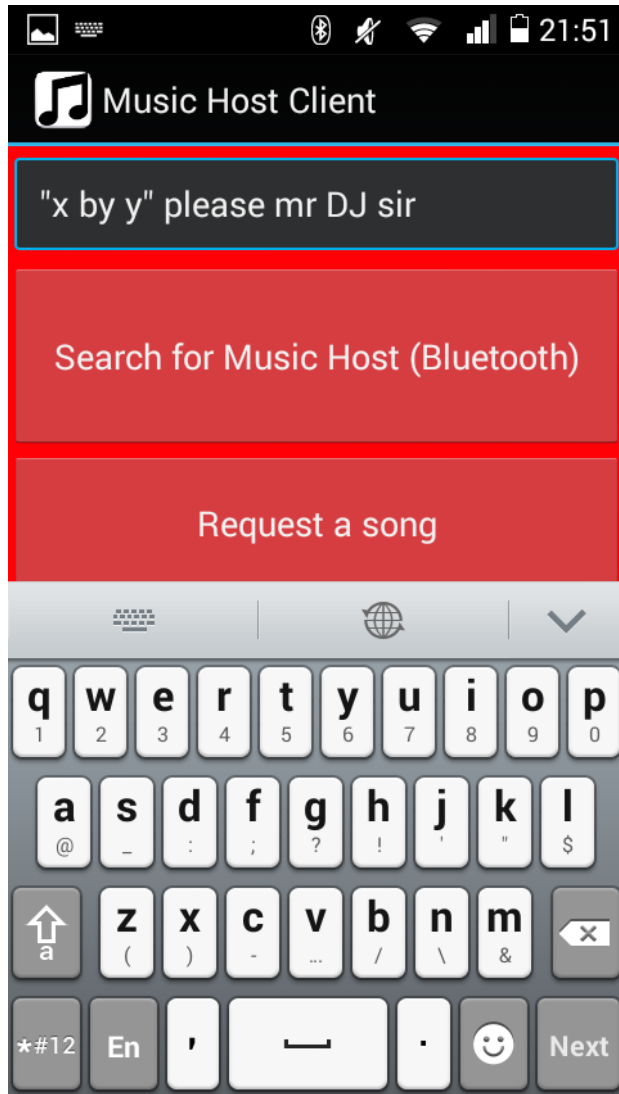


*Message From  
Host*

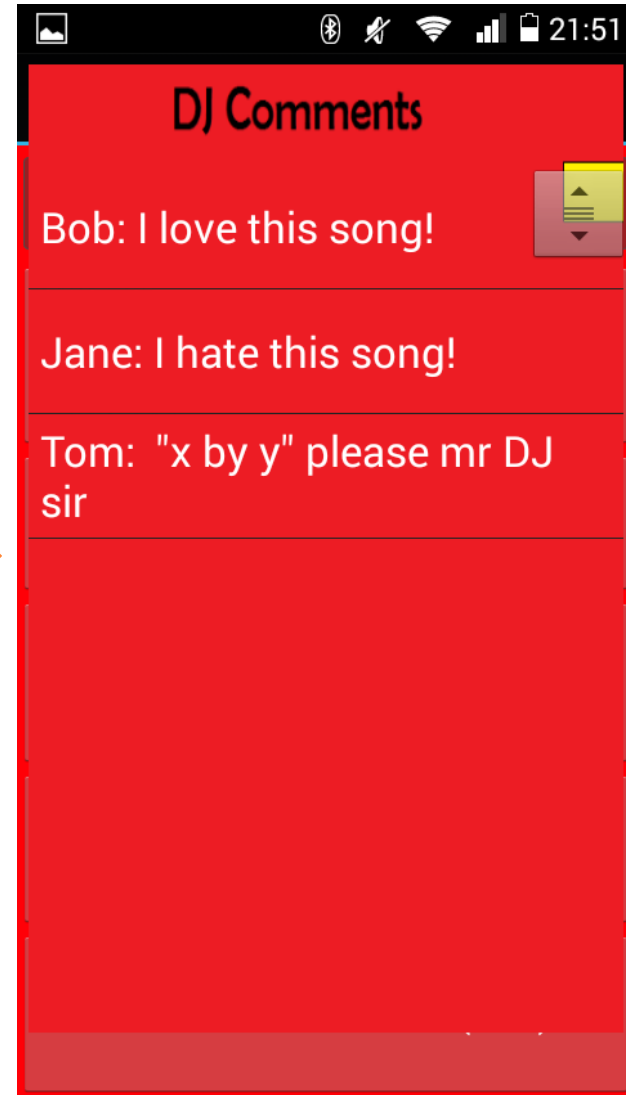


# MUSIC HOST CLIENT

## USE CASE – DJ COMMENT

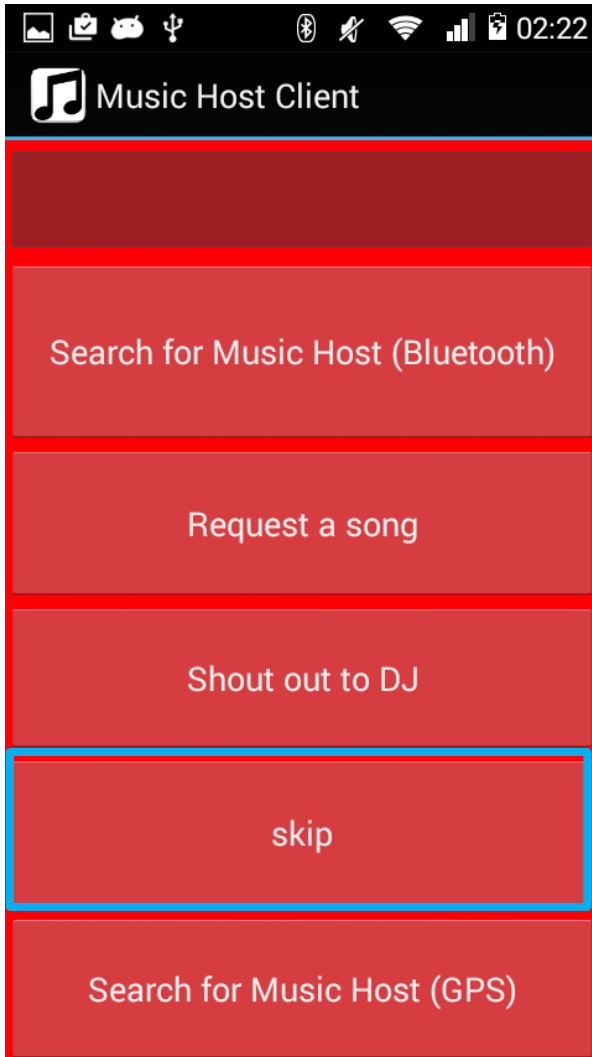


*Button  
Clicked*



# MUSIC HOST CLIENT

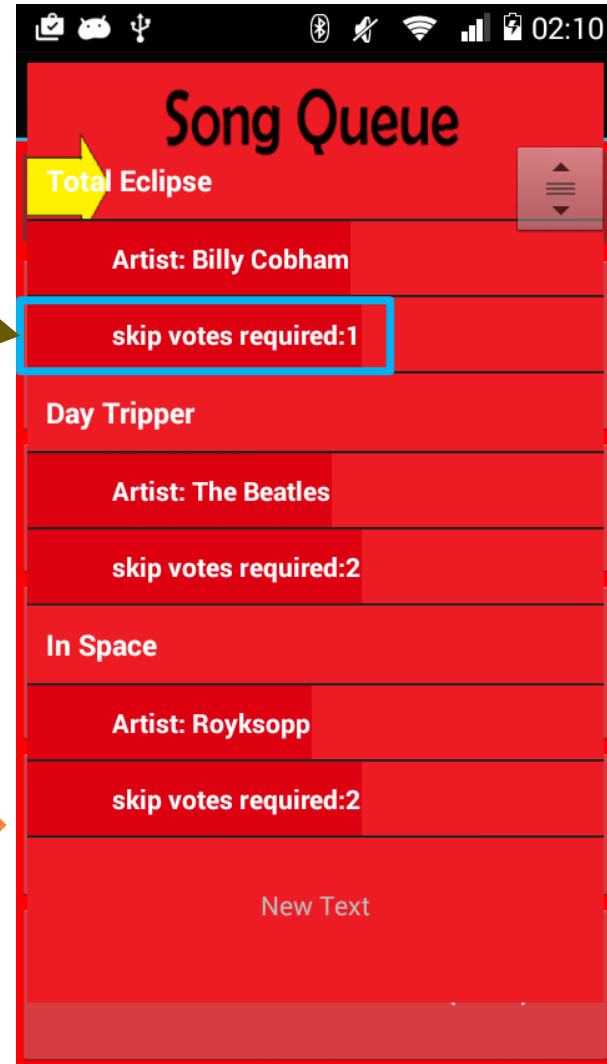
## USE CASE – SKIP SONG



*Skip votes  
required  
decremented*



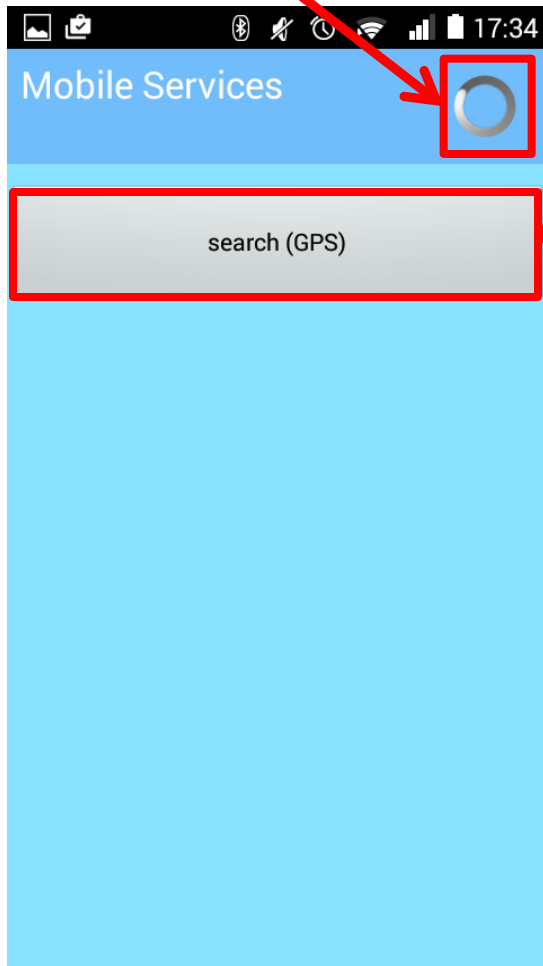
*Button  
Clicked*



# MUSIC HOST CLIENT

## USE CASE – *SEARCH FOR MUSIC HOST (GPS)*

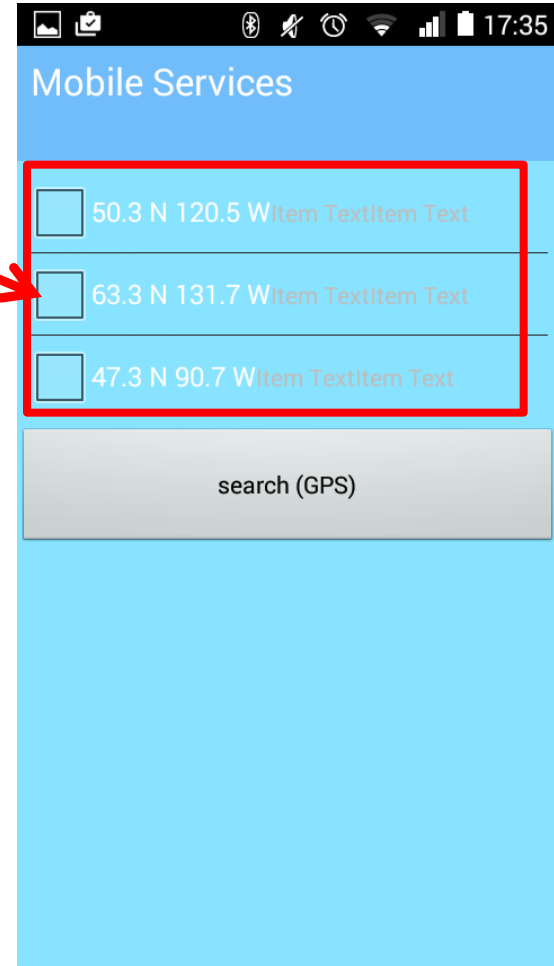
*Progress  
Widget*



*Search  
Button*

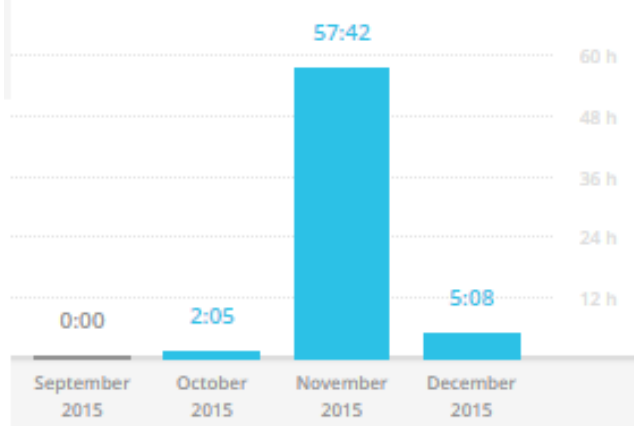
*GPS  
Coordinates*

*Result*



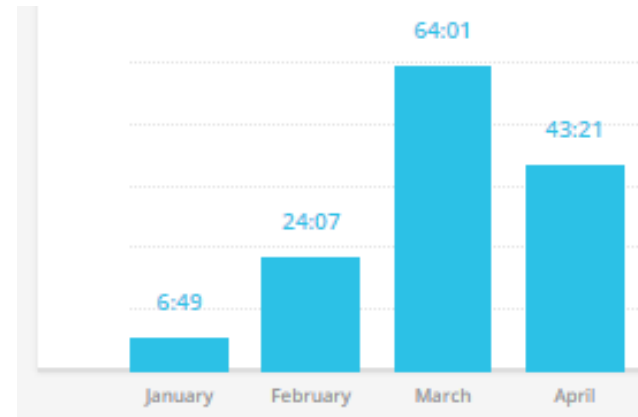
# TIME LOG

## *September - December*

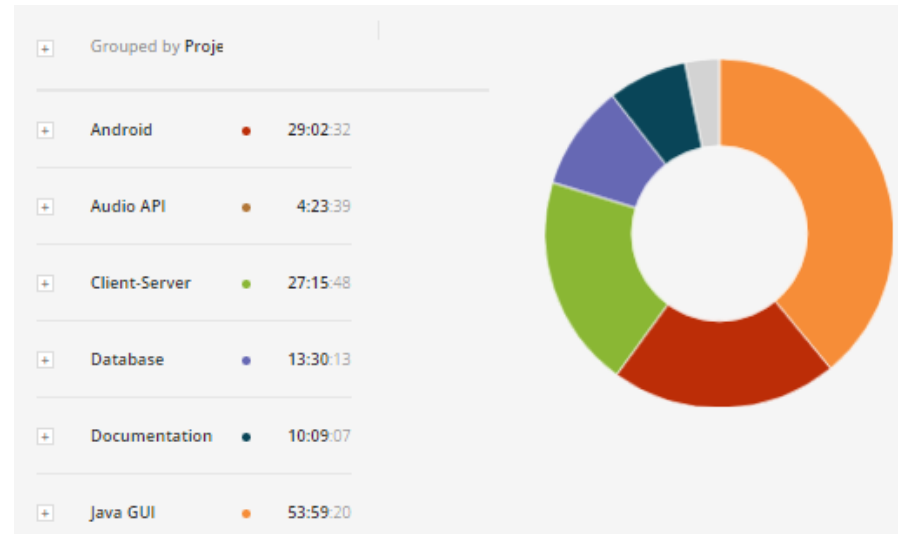
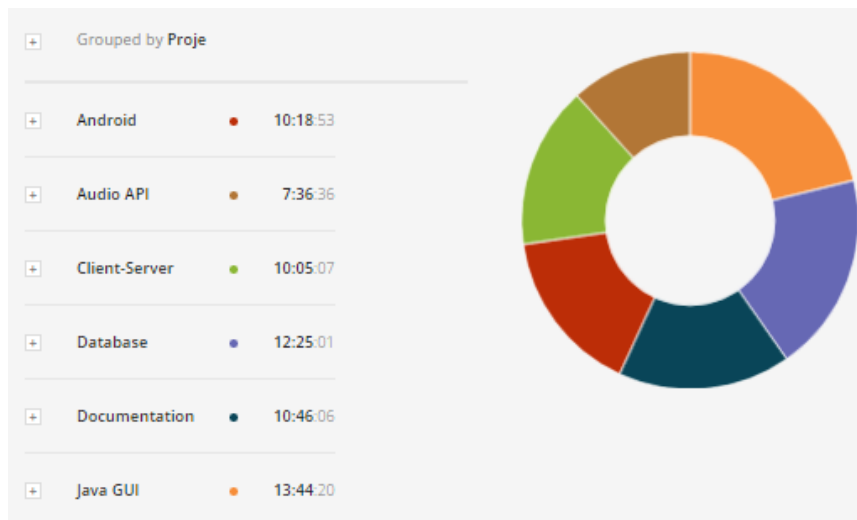


***Total = 64:45:03***

## *January - April*



***Total = 138:20:39***



# CONCLUSION

## ○ Goals

- Demonstration of abilities.
- Develop programming skills.

## ○ Reflection

- More focus on Client/Server Protocol.
- Develop for local song files.
- Develop for Wi-Fi instead of Bluetooth.

## ○ Conclusion

- Project is a product of my abilities and personality.



# REFERENCES

- [github.com/acaicedo/JFX-MultiScreen](https://github.com/acaicedo/JFX-MultiScreen)
- [github.com/marcuspimenta/Chat-Bluetooth](https://github.com/marcuspimenta/Chat-Bluetooth)
- [bluecove.org](https://bluecove.org)
- <https://manage.windowsazure.com/gmit.ie#Workspaces/MobileServicesExtension/apps/MusicHostService/dashboard>
- <https://manage.windowsazure.com/gmit.ie#Workspaces/SqlAzureExtension/SqlServer/muzikhostserver/Database/5.muzikhostserver/Dashboard>

