# Music App for Events (AmpConnect/AmpC)

By

Steve Burns and Evan Mullins

Submitted to
The Faculty of the School of Information Technology
In Partial Fulfillment of the Requirements for
The Degree of Bachelor of Science
In Information Technology





Stephen Burns: _____ Date: 4/18/16

Evan Mullins: _____ Date: 4/18/16

James Scott: _____ Date: 4-18-16

University of Cincinnati
College of
Education, Criminal Justice, and Human Services

April 2016

## Table of Contents



## List of Illustrations

## Abstract

Imagine listening to your music through the speakers of eight different vehicles. What if a DJ never had to carry around his large and heavy speakers that cost hundreds of dollars each or risk breaking these items during transit? This product doesn't replace anything but it makes it better. Using an app, a user connects to a wireless network and starts a party group and listens to their music on their phone. A user's phone can be plugged into an auxiliary port of a vehicle or speaker. A second user connects to that same wireless network/group and sees the playlist of the original user. The second user then plugs his/her phone into the auxiliary port of another vehicle and starts playing the same music that the first user is playing, in sync at the same time. This turns five speakers from one vehicle into ten by adding the second vehicle and so on. There is no more need to keep turning up the volume and blowing out speakers.

## Introduction

Music has become a normal part of our daily lives, we listen to music while on our way to work or school, we listen to music during work or when working out, and we even listen to music while at home relaxing. Music is all around us and sometimes managing our music libraries or coordinating music at events can be a major pain, as well as expensive. "The average cost to install a sound system in a home in 2015 on average was between $248 and $776" according to Home Advisor1, but some cost upwards of $1,000. This price would skyrocket if more than one room was included. For the average user, this price is a little steep and so with our project, we hope to be able to design and create a cost-effective and easy to use alternative that is expandable as well as expendable. We plan to do this by using a Raspberry Pi as a wireless access point and Bluetooth speaker and creating a mobile app to manage a user's music library and create Party Groups. This product will be able to be used just about anywhere as long as there is a power source or mobile device. The scope of the project will be events such as sporting events and parties, and for home use. But, future use and expansions could include schools, offices, malls, and more.

## Project Description

We will develop a phone application so users can connect to a wireless access point or Bluetooth server group and listen to the current music being played by a host. For example: A tailgater wants to listen to their music through their boom box. The volume of the boom box can only go so high. All people listening to the music keep asking for the music to be turned up. We

[1] "Learn How Much It Costs to Install a Surround Sound System." Home Advisor, accessed October 10, 2015. http://www.homeadvisor.com/cost/home-theaters/install-a-surround-sound-system/

have three more boom boxes that are not being used. The other tailgaters will use their phones, essentially connect to the original tailgater's cell, and listen to the current music being played. These other tailgaters can then plug their cell phones into their boom boxes and now we have four boom boxes playing the same music.

The application will be user friendly and allow users to switch control of the music playlist and/or request songs in the playlist. This will also be expandable or modular where you can add more functionality.

## Potential Problems

Currently there are not many solutions available to consumers that allow them to stream their music at an affordable cost. An average sound system in a home for a single room can cost upwards of almost $300 or more as stated previously, adding additional rooms will help see this cost skyrocket. In addition to the cost, installation may require knocking down walls and creating a general mess. With our solution, we want to provide an affordable, expandable, and expendable music server system that will allow consumers to do more than just stream music at home. With our solution, the consumer will be able to stream their own music, allow a friend to stream theirs, and take the application with them wherever they go.

Some of the potential problems that this project may face will be setting up a music application server that takes the users music playlist and replicates it to other devices. We will need to understand how to program and develop a basic mobile app and media server to allow this function. Getting a wireless client to act as an access point to save you the trouble from purchasing an access point might also present itself as a potential problem for the project. To fix

this we will need to find a way to create our own wireless network between devices that will allow other devices to connect just like a wireless fidelity (Wi-Fi) connection. Another issue that may arise would be getting the app to be sharable between users based upon who has master control and who just has listening capabilities. This will need to be done in a way that allows a master user to determine who can share their playlist, but still retain full admin controls. In addition to these potential problems, the user interface may be a challenge for us to create. The user interface needs to be simple and easy to use, but also aesthetically pleasing to the consumer. We will need to research methods on developing usable user interfaces that have a good design as well.

## User Profile

In the figure below (Figure 1.1), we show an example of what a typical core user or "Client" user will be able to do and experience with the music server. This experience should be similar to things the user has already experienced with music streaming (i.e. Pandora or Spotify) but will be a lower cost alternative that they will be able to build upon and customize.

| **User Profile** |
| --- |
| Application: AmpConnect |
| Potential Users<br>Music enthusiasts, members of a household, music DJ's, party goers, shoppers, and students<br><br>(Core Users) |
| Software and Interface Experience:<br>The users would experience the system through an App that connects to a central server. The app would allow for selecting which music to play from the user's device, the server, or possibly another location such as Pandora or Spotify. The app will be designed to be as simplistic and user friendly as possible so that just about any user could use it. The server configuration may need someone who is more versed in computers to set up/change if needed |
| Experience with Similar Applications: |

1. Pandora
2. Spotify
3. Raspberry Pi applications
4. Bluetooth/Wi-Fi

The target user(s) will most likely already be familiar with mobile smart phones and applications and should have a decent amount of knowledge about connecting to wireless networks and playing music. Users with less experience will still be able to use the app but may have some difficulties if they do not understand mobile smart phones or music.

Task Experience:

1. Download music app and connect to wireless access point/server
2. Managing a network
3. Knowledge in music streaming services
4. Selecting music to play and voting on what music to be played next

Frequency of Use:
This will depend on the occasion the user wishes to stream music, this could be a daily use while at home or at school/work, it could be weekly based on tailgating for sport events or parties, and it could be monthly/yearly for special holidays or birthday parties. This will depend on how much the user likes listening to music and where they are located

Key Interface Design Requirements that the Profile Suggests:

1. Simple Interface/easy to use mobile phone application
2. Expected function buttons for music (play, pause, next song, volume up/down)
3. Action buttons
4. Voting system for songs to be played based upon popularity from users
5. Real-time updates and simultaneous streaming of music to multiple devices
6. Similar user friendly interface to other well-known/used music streaming services

**Figure 1.1 – User Profile**

## Use Case Diagram

In the figure on the following page (Figure 1.2), we show the permissions and what each user will be able to do with the functions of the music server. There will be a "Master" or main user who has essentially all permissions and will act as an admin, but also be the controller of the music server. The Master User will be able to control the music, and provision any permissions

to users connected to the access point. This will allow regular "Client" or core users to listen to the music, but also request permissions to share their own playlist if applicable at an event.

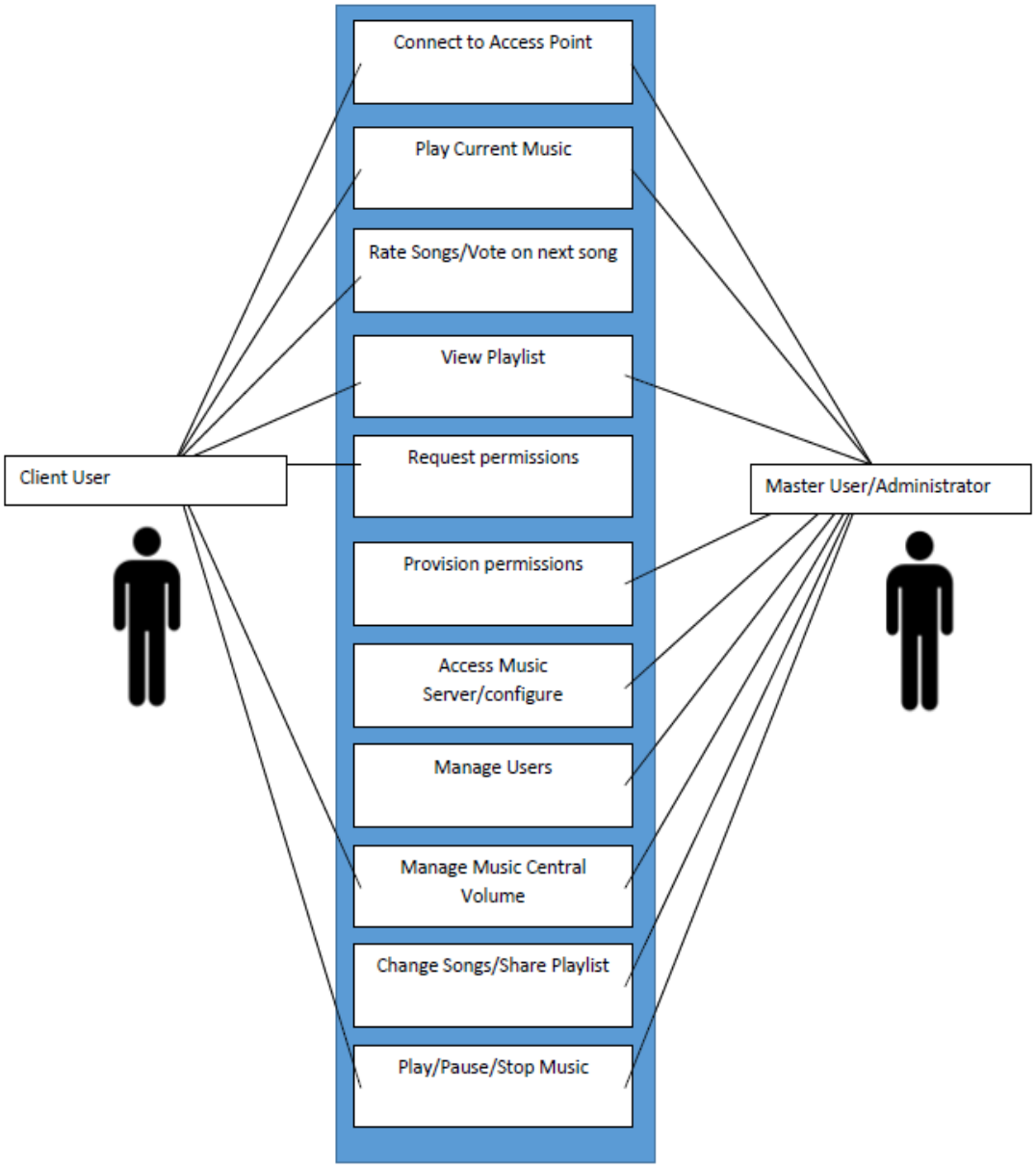



**Figure 1.2 – Use Case Diagram**

## Network Diagram

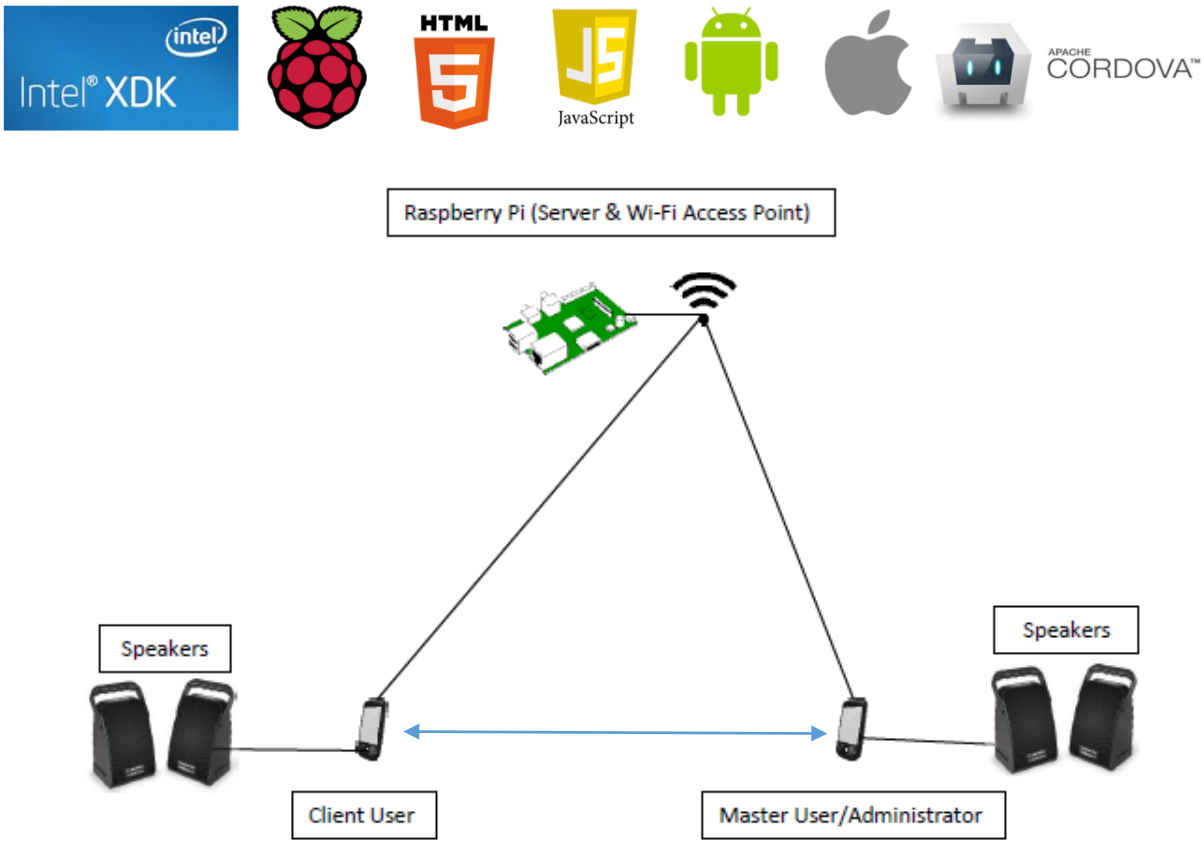



**Figure 1.3 – Network Diagram**

## Objectives/Deliverables

- Develop a wireless network using a raspberry pi and Local Bluetooth Server in order to host an application server to share a music playlist.
- Develop a server that attaches to the user's media playlist.
- Create a phone app for the user interface
- Phone apps will be used to connect to the network/playlist server so the users are able to see the playlist, vote on the songs, and listen to the music being played.

## Proposed Budget

A $70 Raspberry Pi w/ SD card and Wi-Fi adapter have been purchased from Amazon for the project and should be the only hardware necessary. The group has access to old cell phones to use as test phones. The speakers from a vehicle was the original plan for starting this project and will be used as well as PC speakers. The rest of our budget will be searching and/or creating open source programs for the mobile applications. In the figure below (Figure 1.4), we show an example of our costs for the project compared to an average cost a small business might pay in order to create our Raspberry Pi Media Server/Wireless access point and mobile phone application. The costs for creating an application varies based upon the complexities of the application so the price could range anywhere from $3,000 to over $150,000. We also opted to purchase a premium version of a software that has a free version as well. This is GenyMotion which is an emulator for mobile devices (Android, iPhone, Windows Phone) and is much faster than default emulators that come with software packages such as Microsoft Visual Studio. This was an optional cost so we included that below anyway just to show a comparison of just how cost effective our project plan is when we created it instead of a business.

| | Our Costs | Average Costs |
|---|---|---|
| Raspberry Pi (Kit) | $70.00 | $70.00 |
| GenyMotion (Optional) | $136.00 | $412.00 |
| Development Costs | $0.00 | $3,000-$150,000 |
| Installation | $0.00 | $100-$1,300 |
| Labor Hours | $0.00 | $75/hr |
| | | |
| **Total:** | $206 | $3,582-$151,782 |
| | | (*Not including Labor costs*) |

**Figure 1.4 –Budget**

## Software and Hardware Utilized

` We researched different options for both software and hardware and eventually decided upon which software and hardware we wanted to use. For the hardware, we opted for the inexpensive and adaptable Raspberry Pi since it provided more than adequate capabilities for what we were working to accomplish. This was purchased in a kit that included everything we needed to get started (Case, SD Card, Power Adapter, Wireless Adapter, HDMI Cable, and the Raspberry Pi) . For the software, we utilized a number of free and open source applications such as Microsoft Visual Studio 2015 to design the mobile application, Sublime Text 3 for editing text, Ionic & Apache Cordova (Open source HTML/CSS/JavaScript development framework), GenyMotion, and Intel XDK which we ended up using in the end to create our application. Each software allowed us to better design and create our mobile application and Ionic and Apache Cordova allowed for us to create our mobile application easier as both are based upon HTML which we knew and provided a simpler way to create a cross platform mobile application.

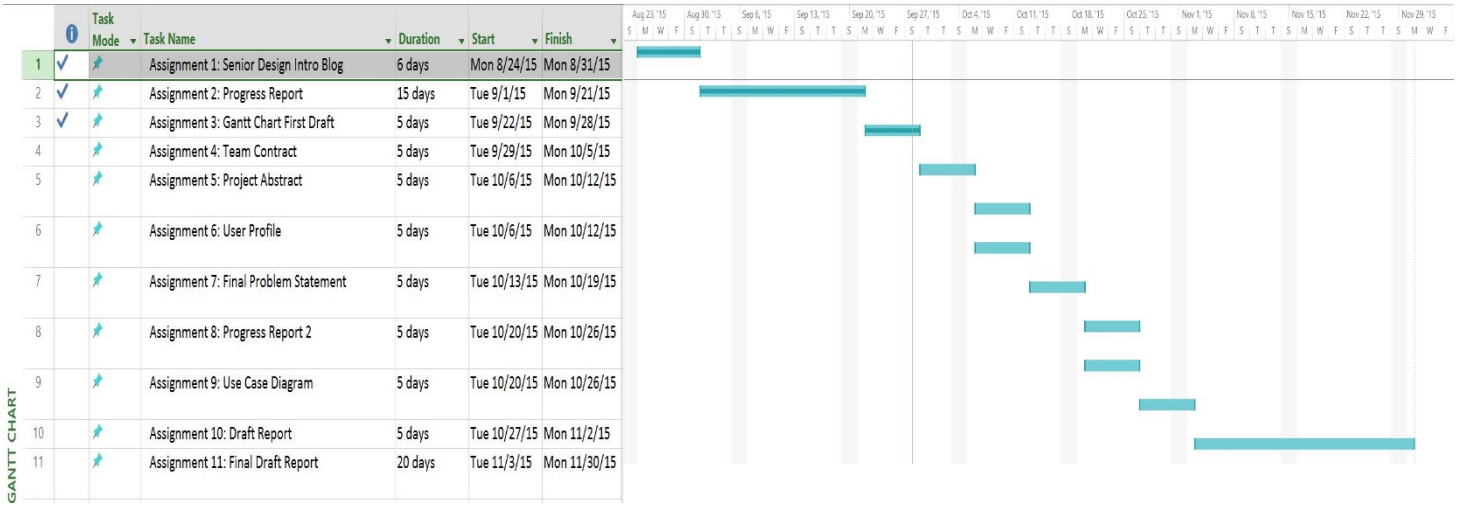
## Timeline & Gantt Chart

For our timeline, we wanted to be able to complete the core of our project by the end of the first semester. We wanted to set up our raspberry pi as a wireless access point and create the foundation for our mobile phone application and have some features added to it before the second semester. We were able to meet most of our objectives and complete our goals that we set for ourselves during the course of the project. We saved some project elements such as customizing the User Interface of our mobile phone application for the next semester so that we could incorporate user feedback into the application. Figure 1.5 on the next page shows a basic

layout of our first phase project timeline and what our goals were for each deliverable. Figure 1.6 shows a basic Gantt chart that follows figure 1.5's timeline and shows a visual aspect of our timeline and when each deliverable was to be complete. Figure 1.7 shows a project timeline and Gantt chart updated for the second semester and what steps we plan on taking. The project timelines will also show what we plan to accomplish at specific date in time.

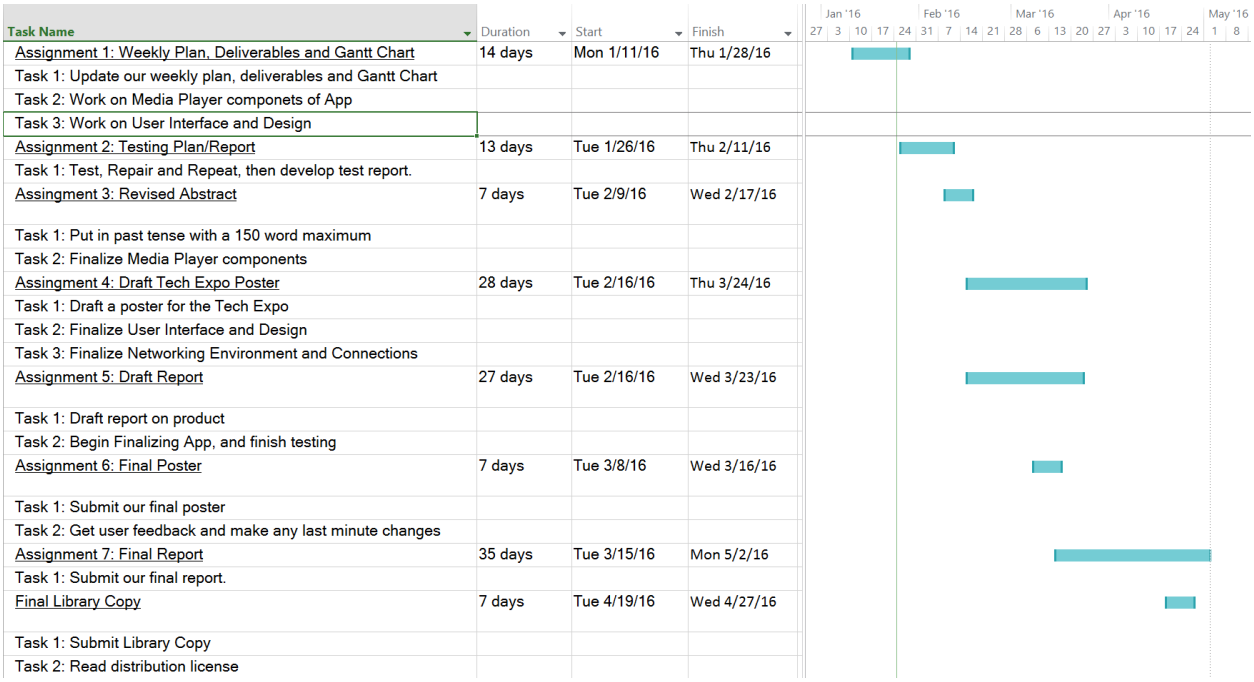
| **Music App and Server for Events** | | | | | 2015 |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| **Group Members:** | | | | | |
| Steve Burns and Evan Mullins | | | | | |
| | Hours (per week) | Start Date | Duration (days) | Due Date | Status |
| **General Deliverables:** | | | | | |
| Assignment 1: Senior Design Intro Blog | 18 | 24-Aug | 7 | 31-Aug | Completed |
| Task 1: Find/Establish Group and Project Idea | | | | | |
| Assignment 2: Progress Report | 18 | 1-Sep | 20 | 21-Sep | Completed |
| Task 1: Seeked Project Approval from Professor | | | | | |
| Task 2: Begin Researching Project | | | | | |
| Assignment 3: Gantt Chart | 18 | 22-Sep | 7 | 28-Sep | Completed |
| Task 1: Meet with Team Members to create Gantt Chart | | | | | |
| Assignment 4: Team Contract | 18 | 29-Sep | 7 | 5-Oct | Completed |
| Task 1: Meet with Team and Assign Roles | | | | | |
| Task 2: Begin Purchasing Hardware for Project | | | | | |
| Assignment 5: Project Abstract | 18 | 6-Oct | 7 | 12-Oct | Completed |
| Task 1: Create Abstact with Team Members | | | | | |
| Task 2: Begin installing Software on Raspberry Pi | | | | | |
| Task 3: Document procedures for Software installation | | | | | |
| Task 4: Setup Server on Raspberry Pi | | | | | |
| Assignment 6: User Profile | 18 | 6-Oct | 7 | 12-Oct | Completed |
| Task 1: Assess information on the user of the Product | | | | | |
| Task 2: Begin researching phone applications | | | | | |
| Task 3: Begin creating Phone app for project | | | | | |
| Assignment 7: Final Problem Statement | 18 | 13-Oct | 7 | 19-Oct | Completed |
| Task 1: Begin testing connectivity between app and server | | | | | |
| Assignment 8: Progress Report 2 | 18 | 20-Oct | 7 | 26-Oct | Completed |
| Task 1: Setup Wireless Speakers/Access Point using Raspberry Pi | | | | | |
| Assignment 9: Use Case Diagram | 18 | 20-Oct | 7 | 26-Oct | Completed |
| Task 1: Further Develop Phone Application and Test | | | | | |
| Task 2: Add additional wireless nodes to server | | | | | |
| Assignment 10: Draft Report | 18 | 27-Oct | 7 | 2-Nov | Completed |
| Task 1: Collaborate with Team and Draft report from research/testing | | | | | |
| Assignment 11: Final Draft Report | 18 | 3-Nov | 27 | 30-Nov | Completed |
| Task 1: Begin prepping for oral presentation | | | | | |

**Figure 1.5 –Project Timeline**




| | Task Mode | Task Name | Duration | Start | Finish |
|---|---|---|---|---|---|
| 1 | ✓ | Assignment 1: Senior Design Intro Blog | 6 days | Mon 8/24/15 | Mon 8/31/15 |
| 2 | ✓ | Assignment 2: Progress Report | 15 days | Tue 9/1/15 | Mon 9/21/15 |
| 3 | ✓ | Assignment 3: Gantt Chart First Draft | 5 days | Tue 9/22/15 | Mon 9/28/15 |
| 4 | | Assignment 4: Team Contract | 5 days | Tue 9/29/15 | Mon 10/5/15 |
| 5 | | Assignment 5: Project Abstract | 5 days | Tue 10/6/15 | Mon 10/12/15 |
| 6 | | Assignment 6: User Profile | 5 days | Tue 10/6/15 | Mon 10/12/15 |
| 7 | | Assignment 7: Final Problem Statement | 5 days | Tue 10/13/15 | Mon 10/19/15 |
| 8 | | Assignment 8: Progress Report 2 | 5 days | Tue 10/20/15 | Mon 10/26/15 |
| 9 | | Assignment 9: Use Case Diagram | 5 days | Tue 10/20/15 | Mon 10/26/15 |
| 10 | | Assignment 10: Draft Report | 5 days | Tue 10/27/15 | Mon 11/2/15 |
| 11 | | Assignment 11: Final Draft Report | 20 days | Tue 11/3/15 | Mon 11/30/15 |

**Figure 1.6 – Gantt chart Timeline**



| Task Name | Duration | Start | Finish |
|---|---|---|---|
| Assignment 1: Weekly Plan, Deliverables and Gantt Chart | 14 days | Mon 1/11/16 | Thu 1/28/16 |
| Task 1: Update our weekly plan, deliverables and Gantt Chart | | | |
| Task 2: Work on Media Player componets of App | | | |
| Task 3: Work on User Interface and Design | | | |
| Assignment 2: Testing Plan/Report | 13 days | Tue 1/26/16 | Thu 2/11/16 |
| Task 1: Test, Repair and Repeat, then develop test report. | | | |
| Assingment 3: Revised Abstract | 7 days | Tue 2/9/16 | Wed 2/17/16 |
| Task 1: Put in past tense with a 150 word maximum | | | |
| Task 2: Finalize Media Player components | | | |
| Assingment 4: Draft Tech Expo Poster | 28 days | Tue 2/16/16 | Thu 3/24/16 |
| Task 1: Draft a poster for the Tech Expo | | | |
| Task 2: Finalize User Interface and Design | | | |
| Task 3: Finalize Networking Environment and Connections | | | |
| Assignment 5: Draft Report | 27 days | Tue 2/16/16 | Wed 3/23/16 |
| Task 1: Draft report on product | | | |
| Task 2: Begin Finalizing App, and finish testing | | | |
| Assignment 6: Final Poster | 7 days | Tue 3/8/16 | Wed 3/16/16 |
| Task 1: Submit our final poster | | | |
| Task 2: Get user feedback and make any last minute changes | | | |
| Assignment 7: Final Report | 35 days | Tue 3/15/16 | Mon 5/2/16 |
| Task 1: Submit our final report. | | | |
| Final Library Copy | 7 days | Tue 4/19/16 | Wed 4/27/16 |
| Task 1: Submit Library Copy | | | |
| Task 2: Read distribution license | | | |

**Figure 1.7 – Gantt chart Timeline Final Semester**

## Future Additions/Plans

Following our project timeline, we knew there would be functions/additions we would want to incorporate into our mobile application. Functionality such as a rating system based upon user voting and what songs in the playlist they would want to be played next was part of what we wanted to be able to add in future phases of our project. Additionally, we also wanted to allow users to create their own playlists and associate songs they own that are stored on their device's local storage. This would allow users a customized experience and provide them the capabilities to create different playlists for different events, for example creating a "Party" playlist for when at house parties to play dance inspiring music or creating a "Tailgating" playlist to play music that gets the fans pumped up for the game. This would allow for a tailored experience for all users of our mobile application.

## Testing

### Overview

Testing is a critical step into ensuring AmpConnect's functionality and bug free environment is in order. This section will explain the testing component of the development cycle for AmpConnect and how certain things were done to complete the testing. AmpConnect is a cross platform compatible mobile application capable of working on IPhone, Android, and Windows Phone but due to time constraints, Android is the platform we focused on for now. The testing section should be used and focused on by:

- Developers
- QA Testers
- Project Managers
- Test Users
- Group Administrators

## Scope

AmpConnect will be tested based upon requirements set by the team members in both functionality and non-functionality requirements (design, color, images, logos).

### Functional Requirements

7. The application should be simple and easy to use
8. The system should allow appropriate permissions of authority while in an Event Group
9. While in an Event Group, permissions/options should vary (i.e. the Group Administrator needs total access while group members only need "read & execute" access)
10. The Event Group Administrator should be able to invite users and set permissions for each user in the group.
11. There should be minimum amounts of screens as possible to avoid confusion while in the application
12. All Buttons should work properly and as intended throughout the application (i.e. all media player options such as play, pause, stop, next, and volume controls should function as intended and expected from users)
13. Users of the application should be able to create their own Event Groups and invite users to the group
14. Music should be allowed to be pulled from the local device as well as cloud based music streaming services such as Pandora or Spotify
15. Streaming and connectivity should work as intended between devices and Raspberry Pi private wireless network should also work as intended

## Test Planning

1. The application should be simple and easy to use
    a. Minimum amount of Buttons on screen at one time
    b. No complex series of screens to jump through
    c. Intuitive design and application layout
2. The system should allow appropriate permissions of authority while in an Event Group
    a. Group Administrator should be able to provide permissions to users to control the playlist/media playing options
    b. Add/remove users from the group
    c. Take ownership of the group
3. While in an Event Group, permissions/options should vary (i.e. the Group Administrator needs total access while group members only need "read & execute" access)
    a. Administrator of the Event Group should have total access/all permissions
    b. Group members should only be able to look at and listen to what is currently playing unless given unique permissions by the group administrator
4. The Event Group Administrator should be able to invite users and set permissions for each user in the group.

**a.** Group Administrator needs to be able to provide unique permissions to each individual user of the group.

**5.** There should be minimum amounts of screens as possible to avoid confusion while in the application.

**a**. There will only be 2-5 screens at max

**6.** All Buttons should work properly and as intended throughout the application (i.e. all media player options such as play, pause, stop, next, and volume controls should function as intended and expected from users).

**a.** All Buttons pushed should provide feedback by default.

**b.** The "Play" Button will play the selected song

**c.** The "pause" Button will pause the current song being played.

**d.** The "next" Button will stop playing the current song and then start playing the next song on the playlist

**e.** The "back" Button will stop playing the current song and then start playing the previous song played.

**f.** The volume control Buttons will control the volume on the local device only.

**g.** The "Connect to Wi-Fi/Bluetooth" Button will prompt the user to select a wireless network

**h.** The "Sort A-Z" Button will sort the music in alphabetical order

**i.** The "Join Group" Button will prompt the user for a music group to connect to

**j.** The “Create Group” Button will start a local Bluetooth server and create a random four digit code used to connect to the group

**k.** The “Music” Button will list all the music stored locally on the device and allow the users to see the playlist on their devices

**l.** The “Home” Button will show the music group the user is in as well as the current song being played at that moment

**m.** The “Settings” Button will have configuration settings for the local device

**7.** Users of the application should be able to create their own Event Groups and invite users to the group.

**a.** A user will be able to create or join an event group

**b.** A user who creates the group will be able to invite other users to join

**8.** Music should be allowed to be pulled from the local device as well as cloud-based music streaming services such as Pandora or Spotify.

**a**. The event group administrator will be able to use music locally from their device or use their cloud-based streaming account.

**9.** Streaming and connectivity should work as intended between devices and Raspberry Pi private wireless network should also work as intended.

**a.** The Raspberry Pi will act as the wireless network

**b.** All users will be able to connect to the Raspberry Pi

**c.** The Event Group Administrator will be streaming the music from his device to all the devices connected to the Raspberry Pi wireless network

## Testing Report

Functional Requirements:

| **Req no:** | **Item #:** | **Test Case #** | **Input** | **Expected Output** | **Actual Output** | **Pass/Fail** | **Reason for Pass/Fail** | **Date** |
|---|---|---|---|---|---|---|---|---|
| 6 | 6.b | 1 | "Play" | Play the song | Nothing | Fail | Programming Error | 2/2/16 |
| | | 2 | "Play" | Play the song | Song played | Pass | Correct Programming | 2/3/16 |
| | 6C | 1 | "Pause" | Pause the current song | Song paused | Pass | The song paused during the current song and picked up where it left off when played again | 2/3/16 |
| **Req no:** | **Item #:** | **Test Case #** | **Input** | **Expected Output** | **Actual Output** | **Pass/Fail** | **Reason for Pass/Fail** | **Date** |
| 7 | 7A | 1 | Create/Start Group | New Group created and started with notification | Nothing | Fail | No notification occurred and group did not start | 2/3/16 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7A | 2 | Create/Start Group | New Group created and started with notification | Group started, but still no notification | Fail | Notification did not occur | 2/3/16 |
| | 7A | 3 | Create/Start Group | New Group created and started with notification | Group started with notification successful | Pass | Group was created and started successfully and notification to user occurred | 2/4/16 |

**Table 1.1 – Testing Process**

## Conclusion

We are just trying to make something better. There are apps in the market that look great and are simple to use. But, that same app limits you to only listening to their music playlists. You're unable to listen to the music stored locally on your phone. Another app allows you to use the music stored locally on your phone but you are limited to fifteen minutes in playing time before you have to provide a fee. We want our app to include all the advantages of other apps, but also provide an ad free and cost free mobile application experience. Our app may have disadvantages due to limitations in technology at this time but with our application, we wanted to provide multiple ways for the users to stream their music, whether that be through Bluetooth, Wireless, or even Cell signal so the users had options. Each outlet has its own advantages and disadvantages but we wanted to allow the user to choose how they stream their music. With Bluetooth the maximum limit of connected devices is just seven devices, with three or four being more practical. With wireless you are typically stuck to a specific area near a Wi-Fi antenna so locations for streaming music can be limited by that. For cell signal, the user will typically have to pay for their data use. Just about everything does have a disadvantage but it's better to have

the advantages outweigh the disadvantages.  That is why we decided to create AmpConnect, a conglomeration of great features other applications incorporate, all into one package. AmpConnect, the mobile application where the user is in control.

# Appendix

Below are a few screenshots of AmpConnect. The first figure (Figure 1.8) shows the main page/home screen which we wanted to keep simple and intuitive for the user. Once the user opens the application they can decide if they want to host a group/create a group or join an existing group if there is one available. Once a group is created as shown below (Figure 1.9), the host is in control and is able to pull music off of their local storage and add songs to a shared playlist that other users can see and access if the host gives them the correct permissions to the group. This allows the host to control what is being played, but also enables the host to allow other members to contribute to the playlist as well.

Other users can join an existing group shown in (Figure 2.1) by typing in the four digit code created when starting a group. This starts a local Bluetooth server on the host device using that four digit code as the connection name so MAC addresses for devices are not shared.


**Figure 1.8 –Home Screen**


**Figure 1.9 –Group Screen**


**Figure 2.1 –Join Group Screen**

The host and only the host of the group has access to the permissions screen, which is where the host of the group can provide unique permissions to the devices attached to the group as shown in (Figure 2.2). When the group is done partying, the host has the option to end the group and this will take every user attached to the Group Ended screen (Figure 2.3). This screen allows users to rate the host of the group and this will provide the host with feedback on whether users liked the group or the songs that were played. We hope to further develop this feature in the future to allow users to check the hosts ratings before they connect so they can determine if the host is a decent host or not. There is still a lot that can be added and developed for our application, but for two networkers developing a software application, we have come a long way.

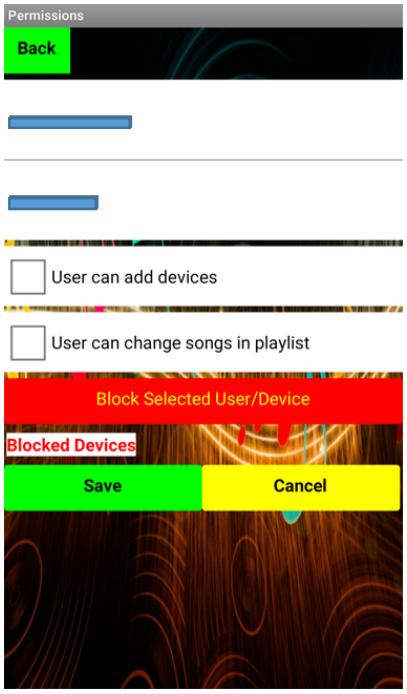



**Figure 2.2 –Permissions Screen**

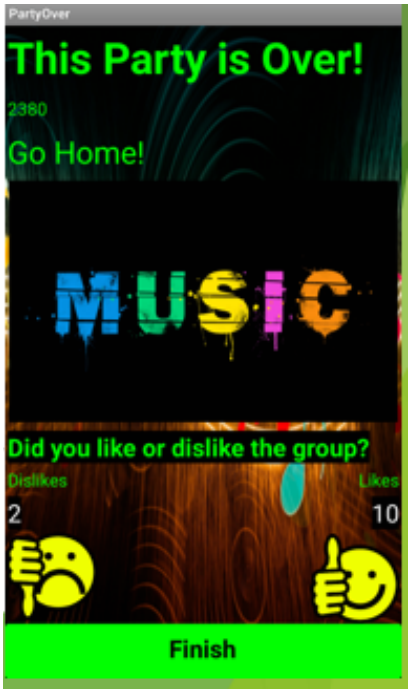



**Figure 2.3 –Group Ended Screen**

# References

1. “Learn How Much It Costs to Install a Surround Sound System.” 2015 Surround Sound System Installation Cost. Accessed October 10, 2015. http://www.homeadvisor.com/cost/home-theaters/install-a-surround-sound-system/