|  |
| --- |
| **Adio** |
|  |
| Sneha Advani, Bharath Jaladi, Arjun Lal, Romit Nagda, Sneha Rampalli  **Team**: 19  **Advisor**: Sangeeta Vohra |
|  |
|  |
|  |
|  |

Abstract

|  |
| --- |
| 000  001  002  003  004  005  006  007  008  009  010  011  012  013  014  015  016  017  018  019  020  021  022  023  024  025  026  027  028  029  030  031  032  033  034  035  036  037  038  039  040  041  042  043  044  045  046  047  048  049 |

This document serves to describe the work done by Group 19 on Adio, an application for Uber and Lyft drivers to play their music during their shifts interspersed with location-based advertisements at a frequency they set.

Motivation & Goal

Advertisers, especially small businesses, want and need innovative and cost-effective ways to reach high-potential consumers. Moreover, consumers want and need relevant and effectual methods to discover and connect with businesses that are easily accessible to them. Rideshare companies, such as Uber and Lyft, provide a unique service and medium for novel, location-based advertising that provides supplemental income to rideshare drivers.

* 1. Market Research

We observed that many Uber and Lyft drivers play music through the radio, which features advertisements at regular intervals that earn revenue for radio stations. In 2019, spending on radio advertisements in the United States is projected to be $17.9 billion. This amount is projected to grow to $18.4 billion by 2023. Furthermore, these number do not include advertising expenditure through music streaming services, such as Spotify and Pandora. However, our team recognized that the majority of radio stations have a wide geographic reach \_\_\_\_. This means radio advertisements

Thus, our team wanted to take this advertising opportunity and improve upon it while also providing another source of income for these drivers.

* 1. Value Proposition

|  |
| --- |
| 050  051  052  053  054  055  056  057  058  059  060  061  062  063  064  065  066  067  068  069  070  071  072  073  074  075  076  077  078  079  080  081  082  083  084  085  086  087  088  089  090  091  092  093  094  095  096  097  098  099 |

Our goal is to provide a platform, Adio, for location-based audio advertising in rideshare vehicles that delivers relevant and effective advertisements between songs played by our rideshare driver-partners. Our driver-partners will be able to download our mobile application, select the frequency of advertisements they wish to play – perhaps one after every two songs – and begin streaming music of their choice from Spotify, Apple Music, or another mobile music streaming service. Based on the advertisement frequency chosen, our driver-partners’ music will be automatically interspersed with advertisements from our advertiser-partners. Using the current location of our driver-partners (and potentially using rideshare trip data), Adio will play geographically relevant advertisements based on target regions selected by our advertiser-partners for each advertisement campaign. The value of Adio is immediate for three primary stakeholders.

* 1. Stakeholders

Our advertiser-partners, many of whom we expect will be small businesses, will be able to use audio advertisements to target geographically relevant customers. The value of each advertisement will be greater than that of a comparable radio advertisement, since businesses can broadcast their advertisements through Adio in a specific geographic region. This means that the cost of advertising through this novel medium will be affordable and practical for a larger group of businesses as compared to radio advertisements. Advertiser-partners will also be able to flexibly create and remove advertisement campaigns and monitor their existing campaigns at any time. If needed, existing campaigns can easily be modified to change the target region, impression limit, or the advertisement itself.

Rideshare passengers who will engage with advertisements through our platform will discover businesses that are relevant and easily accessible to them due to the location-based targeting of our advertiser-partners’ advertisement campaigns.

Our driver-partners will be able to supplement their income through Adio as a portion of the advertisement revenue that we generate will be paid to our driver-partners based on the quantity of advertisements they broadcast to passengers. It is also incredibly easy for driver-partners to begin using Adio, as there is little initial setup or cost required on their part.

* 1. Stakeholder Research

From the inception of Adio, our goal has been to understand the wants and needs of our stakeholders and to best cater Adio to them. Each team member spoke to Uber and Lyft drivers to gauge their interest in our platform and to understand their perspective on features that we planned to include and ones that we had not considered. Specifically, we gave each driver a brief pitch of our idea and asked for their feedback on the following points: (1) whether they play any music while they are driving their passengers (and if so, through which medium) to see if drivers could play advertisements through Adio, (2) if they would use our platform, given the existence of other products with related – but not the same – goals, and (3) if drivers would be interested in beta testing Adio. We received positive feedback from the drivers, especially since the use of our platform requires little setup on their end and there are many potential benefits. The primary concern for drivers was that the passenger experience could be worsened due to advertisements. To alleviate this concern, we allow drivers to control advertisement frequency and believe that the advertisements broadcast by Adio will be relevant and beneficial for riders.

|  |
| --- |
| 100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149 |

We also spoke to family, friends, and other rideshare users to get their feedback on Adio and explore any concerns they may have. The feedback we received was generally positive, especially amongst college students. Given that students on campus frequently use rideshare services to travel to Center City, Philadelphia, they felt that relevant advertisements would inform them of activities and offerings in the area and would definitely be an improvement on traditional radio advertisements. Adults also expressed an interest in this idea, citing business travel to new cities as one of their primary uses for rideshare services. Relevant advertisements would allow them to cost-effectively explore the area around them during their free time.

|  |
| --- |
| 150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199 |

With this feedback in hand, we approached different local businesses and potential advertisers. They were quite receptive to our platform and recognized the financial and location-based targeting advantages of Adio, as compared to other mediums that are available.

Related Work

To understand the existing market, identify competitors, important differentiating factors, what does and does not work, and more, we completed thorough market research. We categorized the existing landscape and competitors as follows.

* 1. Competitors

**Tablet-based Rideshare Advertising**

1. Octopus (<https://www.playoctopus.com/>) provides free tablets to rideshare drivers for passenger use that have trivia games and location-based video advertisements. Octopus pays drivers up to $100 per month.
2. Vugo (<https://govugo.com/>) provides in-car tablet-based entertainment for rideshare passengers in the form of video games, apps, film shorts, sports, and news. Vugo pays drivers $100–$200 per month.
3. Surf (<https://ridewithsurf.com/>) provides tablet-based entertainment for rideshare passengers that includes videos, podcasts, and live radio. Surf pays drivers $0.35 per four-minute advertisement interaction session.

**Rideshare Billboard Advertising**

1. Firefly (<https://fireflyon.com/>) provides large electronic screens that are placed on top of rideshare vehicles and cycle through different geo-targeted advertisements. Firefly pays drivers about $300 per month.
2. Halo Cars (<https://www.halocars.co/>) provides large electronic screens that are placed on top of rideshare vehicles and cycle through different geo-targeted advertisements. Halo Cars pays drivers about $400 per month.

**Music Promotion**

1. Steereo (<https://steereo.com/>) creates playlists for drivers with sponsored music from independent artists paying to promote their songs. Steereo pays drivers once they have reached a minimum of $100 in earnings.

Given the current competition in our target market, we decided to differentiate our product in several ways that we feel are most beneficial to businesses, drivers, and riders. Firstly, our product is purely software-based and thus eliminates any hassle that may come with hardware solutions (such as billboards). Secondly, we integrated with the Spotify API to deliver ads that are interspersed with predetermined playlists set by drivers, which makes the experience more personalized. Thirdly, our solution allows for a passive way of making income for drivers, adding to the profit they make from just driving. Finally, our product will specifically be geared towards smaller businesses who typically do not get the exposure that larger and more well-established franchises receive.

|  |
| --- |
| 200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245  246  247  248  249 |

We believe that these points make our product much more seamless and individualized than other solutions currently in the market. By catering to all stakeholders involved (small businesses, drivers, and riders), our product enhances the user experience on all ends.

Components Completed & In Progress

There are two main components to Adio: a web application, developed with JavaScript, Node.js, EJS, HTML, CSS, Amazon DynamoDB, and Amazon S3, and a mobile application, developed with Swift, SceneUI, and several relevant APIs. At a high level, the mobile application allows driver-partners to sign in and set various parameters regarding the advertisements that they wish to broadcast, such as volume and frequency. In addition, driver-partners can navigate to a dashboard that details their income history over specified timeframes, such as the past month or the past year. The web application is designed for advertiser-partners to easily create audio advertising campaigns that will be seamlessly delivered to consumers via our driver-partners. The website allows advertiser-partners to sign in, upload audio advertisements, and set the center and radius of the circular area in which they desire their advertisements to be broadcast. In addition to creating advertisement campaigns, advertiser-partners can also view their existing campaigns.

Listed below are the specific components that comprise the mobile and web applications, descriptions of their functionality, and our current progress toward these features (**completed** or **in progress**).

* 1. Mobile Application

1. *Initial driver interface.* This is the main screen that allows our driver-partners to begin using Adio during their workday (and to stop using it after completing their rides for the day). It also shows whether or not an advertisement is playing at the current time. **Completed.**
2. *Settings interface.* This view allows driver-partners to calibrate settings, such as the frequency and volume of advertisements that are broadcast, and to turn on and off location sharing. **Completed.**

|  |
| --- |
| 250  251  252  253  254  255  256  257  258  259  260  261  262  263  264  265  266  267  268  269  270  271  272  273  274  275  276  277  278  279  280  281  282  283  284  285  286  287  288  289  290  291  292  293  294  295  296  297  298  299 |

1. *Income dashboard.* This view renders a bar chart detailing the amount of income made by driver-partners through Adio over various periods of time. The graph is displayed using the Swift package Charts. **Completed**.
2. *Interspersing music with advertisements.* This feature automatically pauses the music that is currently playing at specified intervals, plays an advertisement, and then resumes the music. The advertisement is played through an audio player created using Apple’s AVFoundation library. This particular feature is located in the driver interface described previously. **Completed.**
3. *Integration with Spotify.* Our mobile application integrates with the Spotify API to authenticate driver-partners with their personal Spotify accounts and play music of their choosing. **Completed – see appendix for details.**
4. *Playing an advertisement after a set number of songs.* This feature allows driver-partners to specify a number of songs to be played in between each advertisement. The mobile application would connect to Spotify and only play an advertisement once this set number of songs have been played, creating a seamless transition between music and advertisements. **In progress – see appendix for details.**
5. *Integration with Amazon S3.* This component will allow for querying specific audio advertisements from the database that interfaces with our web application. **In progress.**
6. *Driver-partner statistics computation.* This feature will provide calculations and visualizations of various statistics for driver-partners regarding their income and the number of advertisements they have broadcast. **In progress.**
7. *Terms and conditions.* These will detail the relevant legal information and terms of use for Adio, including required driver-partner qualifications. **In progress.**
8. *Geolocation algorithm for ad-selection.* This algorithm will determine driver-partners’ current location and interface with Amazon S3 to compute the most relevant advertisements to be broadcast to consumers, taking into consideration prior impressions for each advertisement and the broadcasting of other driver-partners. In this process, this component will select relevant advertisements and queue them to be broadcast. **In progress.**
   1. Web Application
9. *Account creation.* This allows a new advertiser-partner to sign up with Adio and create an account with an email, company name, first name, last name, and password. **Completed.**
10. *Password hashing with salts.* For security, our platform uses a SHA-512 hashing scheme to store – upon account creation – and verify – during sign in – advertiser-partner passwords on the backend. For each advertiser-partner, we add unique random data – a salt – which we also store on the backend, to their password before providing this combined input to our hash function to guarantee a unique output even when the password inputs are the same. Consequently, the unique hash produced by adding the salt can protect us against different attack vectors, such as rainbow table attacks, while slowing down dictionary and brute-force attacks. **Completed.**
11. *Verification of sign in credentials.* This feature, alongside the previous component, allow for secure verification of email and password combinations during a sign in attempt. In the case of an error, our application notifies the advertiser-partner of the cause for the issue. **Completed.**
12. Map window for geo-location. This allows a user to click on a point of their choosing on a Google Maps API driven map. This click-point serves as the center of their ads’ targeting. Additionally, users can specify a targeting radius here. **In progress.**
13. Upload of audio files. This feature allows users to upload audio files (likely .mp3) that contain individual advertisements. This uploads the advertisement to S3 for later access and stores the ads’ metadata in dynamo db. **In progress.**
14. Evaluation of Completed Components

Below, we list out the various completed subcomponents of our product as well as the methodologies we used to evaluate them.

**Mobile App**

In general, we used a similar protocol to test every feature we implemented on the mobile application. We first tested it on the local XCode iOS simulator, including various device types such as iPhone 11 Pro, iPhone XR, iPhone 8 and iPhone 6. Then, we tested the interface on our devices with different versions of iOS (iOS 9.3, iOS 12.4, iOS 13.2.3). We tested all the features listed in the previous section and attempted various possible scenarios that a user may perform. Our final step of evaluation was to show the application on our phones to possible users and have them interact with the specific feature we were evaluating. Below we talk about specific evaluation points we were looking for.

|  |
| --- |
| 350  351  352  353  354  355  356  357  358  359  360  361  362  363  364  365  366  367  368  369  370  371  372  373  374  375  376  377  378  379  380  381  382  383  384  385  386  387  388  389  390  391  392  393  394  395  396  397  398  399 |

|  |
| --- |
| 300  301  302  303  304  305  306  307  308  309  310  311  312  313  314  315  316  317  318  319  320  321  322  323  324  325  326  327  328  329  330  331  332  333  334  335  336  337  338  339  340  341  342  343  344  345  346  347  348  349 |

**(1)** *Initial driver interface.*We wanted our initial driver interface to be both user-friendly and intuitive. Additionally, we did not want drivers to feel overwhelmed by a cluttered home screen. After a couple of changes to our initial design, we found that potential drivers were happy with an easy-to-use and intuitive homepage that wasn’t too cluttered with extraneous features.

**(2)** *Settings interface.*There are a few crucial points we wanted to touch on here, given that the settings interface is one of the most important views for a driver. First of all, we wanted the settings page to be easy for drivers to navigate to. Secondly, we wanted the settings to encompass everything that potential drivers may deem necessary. Finally, we wanted the settings controls to be easily understandable. The potential users who tried our app appreciated having the ad frequency and volume controls easily accessible, so we moved those to the home page. Beyond that, we added a few more settings, like the options to edit account info and change location preferences to provide additional customizability to the app.

**(3)** *Income dashboard.*Similar to the above, we wanted the profit dashboard to be easy to navigate to and understand in terms of the numbers and figures displayed. We got feedback that just a table of numbers was hard to interpret, so we opted to display a chart of earnings over the year, split by month. Furthermore, we made the figures for monthly and yearly earnings large to draw the attention of the users to the most relevant statistics.

**(4)** *Interspersing music with advertisements.* We wanted the functionality of this feature to follow three main points: (1) background music stops when an advertisement is played, (2) advertisements play to completion and do not cut off before termination, and (3) background music starts back up where it was paused when an interrupting advertisement completes. These points were tested by playing music through Spotify on our devices while Adio was running to observe the functionality of interrupting ads when music is already playing.

**(5)** *Integration with Spotify.*Technical functionality and the user experience here should follow three main points: (1) a driver can connect to his or her Spotify account to play music, (2) a driver does not need to keep logging into Spotify after the first time they do so, and (3) the entire experience should be seamless for a driver and should not halt unexpectedly at any given point in time.

Feedback from users indicated that they would prefer it automatically connected to Spotify when opening the application, so we ensured that the Spotify login functionality could remember passwords if desired after the first login.

**Web App**

**(1)***Account Creation.* We wanted this feature to be as streamlined as possible, given that it is the first thing a user interacts with when he or she uses our app. Thus, we wanted our UI to be intuitive and simple. We asked several friends to look at our home screen and subjectively rate how easy it is to navigate and understand. We received overwhelmingly positive feedback stating that the UI was aesthetically pleasing and that signing up was easy to follow. Objectively, we evaluated this component by ensuring that once a user creates an account, our account database is correctly populated.

**(2)** *Password hashing with salts.*Using sha-256, passwords are hashed, but if a malicious entity figures out this hashing scheme, then every user’s privacy is essentially compromised. To add an extra layer of protection, we created a 16-byte salt that is also hashed in combination with a user’s password, ensuring that figuring out the password hash function is not enough to infiltrate our database. To ensure the correctness of this approach, we observed our databases for consistency. Furthermore, we tried several adverse techniques, posing as malicious entities, to see if we could break our protection scheme, which we could not.

|  |
| --- |
| 450  451  452  453  454  455  456  457  458  459  460  461  462  463  464  465  466  467  468  469  470  471  472  473  474  475  476  477  478  479  480  481  482  483  484  485  486  487  488  489  490  491  492  493  494  495  496  497  498  499 |

**(3)** *Verification of sign in credentials.* Here, we wanted to make sure that our app would correctly check that a user’s login credentials exist in our database. If they do, then the desired functionality is that a user can successfully login. If they do not, then our app simply rejects the user. Thus, we evaluated this by testing login with existing credentials (of an already created account) and non-existent credentials. Using existing credentials, we were able to sign in. Using non-existent credentials, we were denied login, as desired.

|  |
| --- |
| 400  401  402  403  404  405  406  407  408  409  410  411  412  413  414  415  416  417  418  419  420  421  422  423  424  425  426  427  428  429  430  431  432  433  434  435  436  437  438  439  440  441  442  443  444  445  446  447  448  449 |

**(4)  Map window for geo-location.** Our map right now correctly loads on the advertiser/business side within the web application. To fully evaluate its current functionality, we wanted to make sure that interacting with it successfully selects the latitude and longitude points that the advertiser chooses. To this end, we tested several selections on the map (just clicks) to see if the selected latitude and longitude pairs are populated. We tested throughout the map’s viewable area, including points that are within near proximity to each other. Through this, we were able to see that interaction worked correctly.

1. Demo

Below is a link to a video demo of both our mobile app. It shows a background song playing, a 15-second audio ad playing, and then the background music resuming while the user explores the other pages and features in the app.

<https://youtu.be/K8ysW92lypA>

See **Appendix B** for more screenshots of both the mobile and web application.

1. Conclusions and Future Work

Our current implementation of the mobile and web applications provides all of the critical functionalityof our desired product.Next, we need to iterate on the current implementation, finish integrating the mobile and web applications together seamlessly, and add any additional features that may be useful to the drivers and advertisers, such as dashboards to track ad campaigns and additional metrics for drivers to track ad income.

We plan to iterate on the existing features by receiving more feedback from actual Uber/Lyft drivers and potential advertisers, particularly focusing on the evaluation points mentioned in Section 4. We will finish up the integration of the mobile and web apps by working around a Spotify API bug we are facing (described in more detail in Appendix A) and integrating the database of audio ads and target locations for the campaign into both the web and mobile application. This will allow us to intelligently pick an advertisement to be played on the mobile app that targets the current location.

Additionally, we would like to add more features that allow advertisers and drivers to track their progress and revenue, and we would like to implement additional security features. Since our app has the potential for drivers to earn money simply by playing audio, we would like to ensure that this opportunity will not be abused or scammed. We will incorporate security checks, such as checking whether a driver is currently driving at a particular speed, integrating with the Uber or Lyft API to check whether a driver is currently on a trip, and ensuring that the phone’s volume is at an audible level while ads are playing. Once these features are implemented, we will have a much more robust evaluation plan.

When all our features are completed and finalized, we aim to partner with some local business in the Philadelphia area to test our ad services with real ads, as well as with some Uber and Lyft drivers in the area who are willing to test our app while they drive. We hope to receive positive feedback from drivers, riders, and advertisers after our app is completed, as we believe Adio has the potential to benefit all stakeholders involved in an extremely positive way.

1. Acknowledgements

We would like to thank Professors Ani Nenkova, Sangeeta Vohra, Zachary Ives, and Clayton Greenberg for their help and insights into how to move our project forward.

References

<https://www.playoctopus.com/>

<https://govugo.com>

<https://ridewithsurf.com/>

<https://fireflyon.com/>

<https://www.halocars.co/>

https://steero.com/

1. Appendix
   1. Appendix A: Technical Challenges

The biggest technical challenge that we have faced so far involved integrating the Spotify API into our Swift code for the rest of our application. We discovered that the issue we are currently facing is due to a recent update in iOS 13.0.0+, which is detailed below.

When we first integrated with the Spotify API, we were able to successfully import the package into our project and connect to Spotify.com to allow the user to authenticate (or redirect the user to the Spotify app on their device and successfully authenticate them). In our demo video and screenshots, we showed this correctly working functionality of logging in through Spotify.com on the XCode simulator. However, when trying to access properties of the current song playing on Spotify, such as the current title or the duration of the song, we ran into errors that prevented us from reading this information from the Spotify API. Because the XCode simulator is not able to install the Spotify app (only physical devices can), our demo video and screenshots do not show the exact issues we ran into when accessing Spotify information during ad playback on our physical devices.

|  |
| --- |
| 500  501  502  503  504  505  506  507  508  509  510  511  512  513  514  515  516  517  518  519  520  521  522  523  524  525  526  527  528  529  530  531  532  533  534  535  536  537  538  539  540  541  542  543  544  545  546  547  548  549 |

As a replacement for the intended functionality that integrates with Spotify, the current version of our mobile app sets a timer that maintains a set amount of time between each ad. Every few minutes, the app is able pauses the currently playing source of music, plays our ad, and continues playing the original of music. While it meets our evaluation plan for pausing/playing music, this implementation does not take advantage of the additional information that the Spotify API would provide us, such as the duration of the song and the amount of time remaining in the song). Ideally, we would like our app to utilize the information sent back by the Spotify API to determine the end of each song, which would allow us to seamlessly transition between songs and ads without any harsh interruptions. We have written out the code for doing this, but this functionality is currently blocked because of the errors we ran into when accessing Spotify track information after authentication.

|  |
| --- |
| 550  551  552  553  554  555  556  557  558  559  560  561  562  563  564  565  566  567  568  569  570  571  572  573  574  575  576  577  578  579  580  581  582  583  584  585  586  587  588  589  590  591  592  593  594  595  596  597  598  599 |

After further researching these errors and speaking to others with experience using the Spotify Swift API, we discovered that the source of our errors was a new bug introduced with iOS 13.0.0+. We hope that this bug will be fixed in a future iteration of iOS that will be released soon, but in the meantime, we have found a workaround that we will try implementing that allow us to access the Spotify API in iOS 13.0.0+ without any errors.

Our immediate next steps for the mobile application involve attempting to resolve these errors using the workaround for the Spotify API that we found online. Despite these challenges with the recent iOS release, we expect to have completed all of the Spotify integration and pausing/playing features very soon, once we resolve this error.

* 1. Appendix B: Screenshots of Mobile and Web Application

|  |
| --- |
| 600  601  602  603  604  605  606  607  608  609  610  611  612  613  614  615  616  617  618  619  620  621  622  623  624  625  626  627  628  629  630  631  632  633  634  635  636  637  638  639  640  641  642  643  644  645  646  647  648  649 |

|  |
| --- |
| 650  651  652  653  654  655  656  657  658  659  660  661  662  663  664  665  666  667  668  669  670  671  672  673  674  675  676  677  678  679  680  681  682  683  684  685  686  687  688  689  690  691  692  693  694  695  696  697  698  699 |

A screenshot of a stereo

Description automatically generated A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated A screen shot of a smart phone

Description automatically generated