WolfPool: An Easy way to plan your rides

Shyam Katta
Computer Science
N C State University
skatta3@ncsu.edu

Santhosh Dharmagadi
Computer Science
N C State University
sdharma@ncsu.edu

Sai Harsha Suryadevara
Computer Science
N C State University
ssuryad3@ncsu.edu

Sai Sri Harsha Kanamanapalli
Computer Science
N.C State University
skanama@ncsu.edu

ABSTRACT

Most of us fancy to commute at a cheaper cost. Discerning the rapid growth of cities and connectivity, one can clearly notice the limited connectivity of public transportation and other transport services to certain places. There is a need to effectively manage the transportation as well as travel costs while having minimal impact on the environment. Today, we have several applications such as Uber, Lyft which provide transportation services but is there an application which allows the users to pre-book a ride to a certain destination on a particular date and selected preferences and travelers, with a pre-estimated cost? The aim of this project is to create such an application which provides the users a platform to pre-book a ride and commute to different places in a cost-efficient manner with selected riders. This paper discusses the idea of the application, the feasibility of the application and the basic features that will be provided in the application.

1. INTRODUCTION

Who does not like to avoid public transport while commuting to different places in a cost-efficient way? Nowadays, car sharing has become so common that everyone prefers to share the ride, cost, and travel comfortably. There are applications like Uber and Lyft which provides transport services and ridesharing but what if there is an application which lets the user communicate with other users in the application and share a ride with them to a particular destination? Will it be easy for the users to ride with known people in a cost-efficient manner? As this is not a commercial organization, the cost for each trip will be equally split for the cost of the ride. What if the user can also book the ride well in advance of the travel date and have multiple rides booked in advance? Will this be helpful for the user who travels to different locations on a daily basis?

WolfPool is an application which allows the users to commute to different destinations by sharing the rides with known people in a cost-efficient manner. The users can register in the application and search for the plans already existing and join the plan if the conditions set by the users in the ride matches or the user can always go ahead and make a new plan by talking to other users. The user can get a vehicle and make a plan or the user can join the plan as another rider. This application also lets the user have an estimate of how much it's going to cost for that trip and the users can pay for different options one being virtual wallet option. WolfPool ensures complete safety of travellers by limiting the access to the application to only university affiliated personnel. A survey based on feasibility of the application and different features of the application was given out and the results were positive and the feasibility of the application was accepted.

2. LITERATURE REVIEW

Mobility is one of the most basic features of our modern society. People and goods move around the entire Earth in a continuous and broad attempt to fulfill economic, safety and environmental goals. The Mobility Management or Transportation Demand Management is a collection of strategies for encouraging more efficient traffic patterns towards achieving specific planning objectives [1]. For example, people can choose to switch from peak hours to a non-peak time, or to cycle instead of using the car. Administrative regulations could introduce incentives or reimbursements when alternative commuting modes are used [1]. Governmental policies could include fuel tax increases or pay-as-you-drive freeway taxes or car insurances. The applications integrate diverse computing languages with platforms, standards, and technologies.The experimental results are encouraging, allowing us to consider that seamless integration of hybrid management systems for transportation could have tremendous economic and social impact on a global scale [2].

Gathering people into common trips leads to individual and social efficiency. At a personal level, it reduces the total traveling cost and the driving stress as well.Although it is less comfortable than using the personal car and people usually need more time for performing the travel, the broad acceptance of the shared-use. From the social point of view, less fuel consumption, less CO2 emission, less traffic congestion and more social interaction are the benefits of sharing cars or vans [3]. By saving natural resources like oil, by producing less pollution and by cutting the travel expense, using a car for many persons is an advance to sustainable transportation systems. As an alternative to public transportation and taxi services, sharing a car combines the benefits of a shared cost with the flexibility of a taxi ride.The main idea of shared-use mobility is to share the transporta-

tion cost between multiple participants [4].

The number of transportation booking applications have exploded over the last few years primarily due to the increase in GPS enabled devices, fast and reliable mobile connectivity.These apps offer real benefits to customers and companies and location plays a critical role for all of them. Google Maps APIs enable many feature-rich capabilities that can provide the best location-based information and user experiences for booking and tracking the perfect ride. Google Maps APIs are available for Android, iOS, web browsers and via HTTP [2]. All of the services allow you to embed maps within your application on each platform and place markers on top of the maps showing the location of any customers or vehicles. Using the Google Maps API for your customer-facing applications may be the first start to incorporating intelligent location into your business. [3].

3. SOFTWARE ENGINEERING

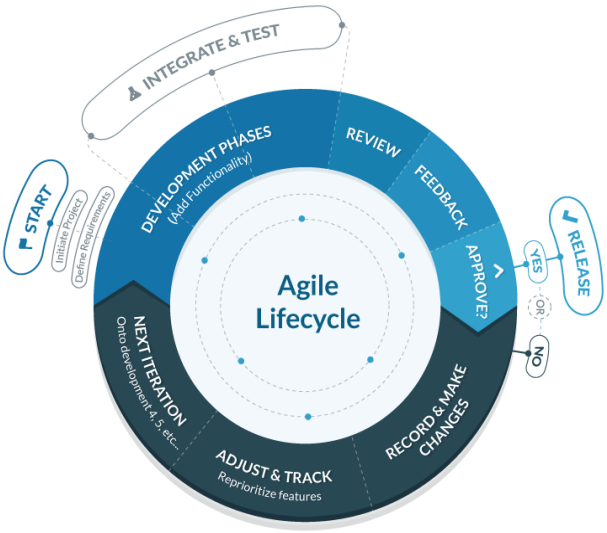

Figure 1: Agile Methodology

3.1 Software Engineering Methodology

We followed Agile Methodology in developing our project since the project was supposed to be delivered within a stipulated timeline. One person among us(rotational basis) acted as a nominal end user of the client and we gathered requirements from the nominal client. Once the requirements are gathered, we split them into functional non-functional requirements. As there was a strict bound on project timeline we only picked up very few important requirements and converted them into user stories.

Few successful **user stories** are, 1. As a user, I can login into the application so that I can browse through my dashboard 2. As an enterprise, I can log in and upload advertisements of my choice with description. As an enterprise, I can view stats on a visual model.

We even added few **technical stories** in line with user stories. Few of successful stories being 1. As a NodeJS developer, I should receive rest API calls for every user request made from front end 2. As an AngularJS developer, I should receive corresponding data to the rest API calls made in a correctly formatted response, so that I can render them correctly.

There are weekly deliverables of the project, we clearly distributed the user stories among ourselves in line with the roles assigned.

3.2 Problems

Better, Faster, Cheaper: Most of the software projects face this kind of problems, no exception is our project. Given the time bound and cost limits, we opted faster and cheaper among the three. When we chose this model, we are prepared for more defects. We tried to deliver the majority of the features and the results were in line with our assumptions. Choosing these both, we were forced to do more fixes than expected. The same is reflected in our pull requests.

3.3 Strategies

We chose **Don't estimate method** as we have very less time. Instead of spending time on effort estimation for the project we thought, investing more time in development and fixes.

Does this mean we haven't made any estimations?

No, we do really estimate but with many small bangs. Better user stories and intuitive deliverables made it easy in developing this project.

3.4 Code Review

Given the rotational change of roles for each of team members, we were able to come up with code reviews throughout the project. As a result, the code became more efficient and minimal when a pair programmer pointed out the issue, alternative methods. When a nominal user points out the expectations of a normal user and when the technological expert speaks about the limitation. The commits made to the master branch were authorized by other team persons which resulted in minimal issues in the master branch.

3.5 Testing

The user stories were tested at the end of each user story by the nominal user. The pair programmers **regression tested** the code regularly after each major update. Availability of end user at end of each **sprint**, **pair programming** of the developers working on related features and **code review** of code before each commit to master branch helped us to overcome many issues faced by our don't estimate approach Which in turn minimized the total testing need to be done. During development, each developed ensured that the individual modules worked as intended. Apart from this manual testing is also done at before each sprint.

3.6 Pair Programming

After spending some quality time on debugging the project individually, we realized that debugging the application in teams of two might give better results and we employed pair programming methodology to get started with the wolf pool application. This approach helped us in finding and

resolving the bugs in the source code quickly. Also, working together as a team helped us in reducing the mistakes that the developers might commit otherwise.

4. SYSTEM ARCHITECHTURE

4.1 NodeJs

The backend for this application was implemented in NodeJs. NodeJs is an open source, cross-platform Javascript runtime environment for executing JavaScript code server-side. The reason for choosing nodeJs for the backend is its performance due to its event-loop. It uses an asynchronous event-driven model and is designed for writable scalable internet web applications, notably web servers. Moreover, one more reason to choose Node.JS is that it can help them write applications both in server-side and client-side and it becomes much easier to hold on with JavaScript for both sides of the client and server. One more reason to choose nodeJs is its ability to balance load at a time due to its event-loop nature when multiple users hit the server. The frontend of the application is also written in nodeJs using handlebars which use the express-handlebar module to render the javascript code. Moreover, APIs like Uber API, Lyft API, Splitwise API and google maps API are supported by nodeJs which was the driving force to develop the application in nodeJs.

Why MEAN Stack and not LAMP STACK?

One differentiating feature of the MEAN stack is that JavaScript code can be reused for both client and server, and makes life easier for developing teams to switch between the two. It makes the workflow homogeneous and also enables two groups of full stack developers to work well together on the application as a whole. On the other hand, the LAMP uses PHP/python for server and JavaScript for a client which helps maintain security if the client and server code bases are kept separate. Another feature of MEAN is that it uses a Non-Relational database while LAMP uses a Relational database which pro- videos development team with the benefit of enhanced speed for data retrieval.

4.2 Middleware - ExpressJS

The middleware was implemented using ExpressJS. ExpressJet a prebuilt NodeJS framework that can help you in creating server-side applications faster and smarter. Simplicity, minimalism, flexibility, scalability are some of its characteristics and since it is made in NodeJS itself, it inherits itsperformance as well.

4.3 Database - MongoDB

MongoDB is an open source document-oriented NoSQL database. Instead of storing the data in tables made out of individual rows, like a relational database does, it stores the data in collections made out of individual documents. In MongoDB, a document is a big JSON blob with no particular format or schema. MongoDBs document data model maps naturally to objects in application code, making it simple for developers to learn and use. Documents give you the ability to represent hier- archical relationships to store arrays and other more complex structures easily. MongoDB was preferred as it supports different structures to be stored at once.

Why MongoDB and not MYSQL?

MongoDB enables us to build applications faster, handle highly diverse data types, and manage applications more efficiently at scale. MongoDB can also be scaled withinand across multiple distributed data centers, providing new levels of availability and scalability previously unachievable with relational databases like MySQL. As the deployments grow in terms of data volume and throughput, MongoDB scales easily with no downtime, and without changing the application. In contrast, to achieve scale with MySQL often requires significant, custom engineering work. MongoDB has a rich query language, highly-functional secondary indexes (including text search and geospatial), a powerful aggregation framework for data analysis, faceted search, graph processing and more.

4.4 Amazon Web Service

Amazons Elastic Beanstalk which is an easy-to-use service for deploying and scaling web applications and services. Elastic Beanstalk automatically handles the deployment, from capacity provisioning, load balancing, autoscaling to application health monitoring. For the backend framework, we are using Node.js. 64bit Amazon Linux 2017.09 v4.4.4. The Instance type is t1.micro and availability is in all zones provided by AWS. Our application s primary hosted region is the North East US region in Virginia but it is loaded balanced and can instantiate new hosts in multiple regions.

Why AMAZON Web Service?

We compared the services offered by Google Cloud Platform and Heroku. We also decided to finally go with was because it provides a lot of configurability and control over the platform we use to host our application. Due to the extensive documentation, support, and experience of working with AWS, it was the obvious choice. The options for scalability in terms of processing power and availability by means of multiple server zones are varying and can be selected at a very fine level based on the cost and efficiency.

4.5 MailJet

MailJet is a Paris-based, all-in-one Email Service Provider(ESP) that allows businesses to send both marketing and transaction email. Initially, we tried sending email from a Google Gmail account but it was not supported in AWS so we had to change to Amazon Simple Email Service (SES)but it had a limit on the number of emails that can be sent using that service. To increase the limit, we would have to increase the quota which would incur additional charges.So we decided to use Mailjet, which is a free service that is heavily used in our application.

5. WHY WOLFPOOL

There are few questions about the uniqueness of this project. Let's go through them.

Why Wolfpool?

Wolfpool is an application mainly focused to provide a platform for the users to make plans and travel with known people in a cost-efficient manner.

What is new in Wolfpool when compared to other transport service applications?

- Wolfpool provides an option to plan multiple trips far advance in time compared to book a single trip in shorter time for Uber pool.
- The users in wolf pool can share the ride with known people as compared to strangers in Uber/Lyft pool.
- WolfPool is a cost-efficient application as the cost of the trip can be equally shared among a maximum number of people.
- The users can chat with other users in the trip to settle on common terms over various preferences.
- The user can decide to join a trip if the user is satisfied with the ratings of other users.

Are the users secure when they travel to different destinations?

- Wolfpool lets the users register only if they have a .edu email id.
- Wolfpool provides an option of sending notifications to the registered contacts when the user presses the panic button.
- The users choose whom to ride with instead of traveling with strangers.

6. FEATURES

Wolfpool is a web application which lets the user share rides with other users in the application by talking to them and lets them plan the trip ahead. The user can register in the application and input the travel details like source address, destination address, travel time and travel date and search for an existing plan. If there is any match to this trip, the similar plans will be displayed to the user if there is any vacancy in the trip and the user can join in any of the trips. If there are no trips available, the user can create a new plan and let others join the plan. Few new features have been added to application which lets the user select the most appropriate trip to that user and decide which plan to join based on few results like price estimation and check which plan is most economical. Also, WolfPool is designed keeping campuses and students in mind for security and safety concerns. Considering the disadvantages of other platforms, the features provided by WolfPool would be an ideal way to plan rides.

6.1 Security

WolfPool ensures complete safety of travelers by limiting the access to the application to only university students and only users with legitimate email ids can use the application.This will be performed by parsing the email address of registrants to look for .edu domain. Due to this, the system is accessible to members of the university whose identities are verified and trusted by the university itself. As a result of this, intruders are restricted to use our application Moreover, the users can choose whom to travel with which ensures more safety.

6.2 Additional Search Options

The basic search matches the trip if the source address, destination address, trip date and trip time match and if there are any vacant seats on the trip. Few additional parameters like gender preference, luggage, and minimum rating options are given to the users while searching the trip. This lets the users choose the kind of people they want to share a trip with and as everyone cannot get as many numbers of bags they can, a limit of 4 bags in applied to each trip and the luggage option makes sure that the luggage of the users is managed properly without any conflicts. There is a chat room for each trip where the user can resolve conflicts if any.

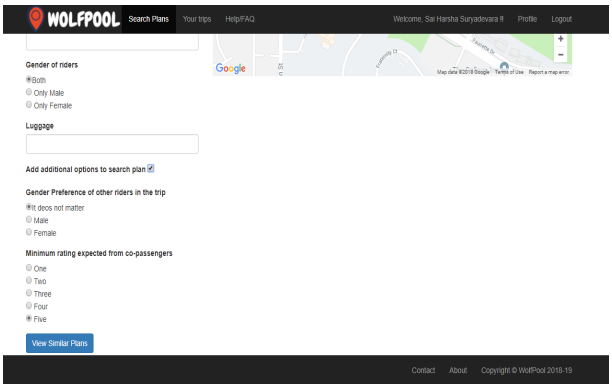

Figure 2: Additional Search Options

6.3 Expense Tracking

Expense tracking is a feature which lets the users split the bills proportionately with the other users who shared the trip. We can also select only the users between whom the expenses are to be shared. If any user pays the trip, the user can add this transaction in the expense tracking page and the proportionate cost per user will be updated in the database and will be reflected in other users' accounts who shared the trip. Along with the expense tracking feature, a virtual wallet has been implemented which stores the transactions and the amount the user owes or is owed. The average cost per rider in the trip is obtained from the price estimate option provided in the get price estimate feature which will be helpful to get an estimate of how much each user will have to spend.

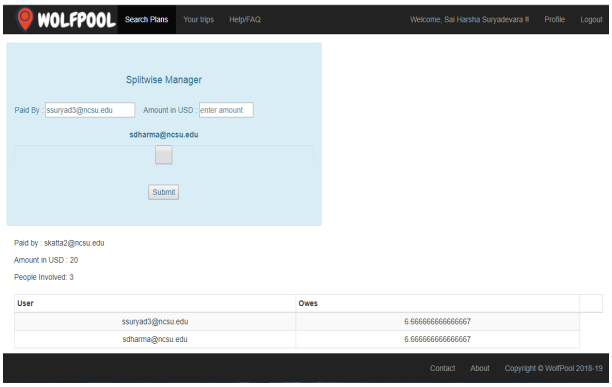

Figure 3: Expense Tracking

6.4 Trip cost Estimation

When a user searches for a trip, the user is given an option get to know the estimate of that specific trip and then decide whether to join the trip or not. The estimate is provided by using Uber and Lyft APIs. When the user clicks on the get Estimate button for a specific trip from the list of trips returned from the search query, the user gets a list of cabs available in Uber and also a list of cabs that are available in Lfyt. Other details like minimum cost estimate, maximum cost estimate, duration, distance and the average cost for each rider in the trip are also displayed in the table. If Uber or Lyft is not providing service for the selected source and destination selected then nothing is displayed and empty tables are populated.

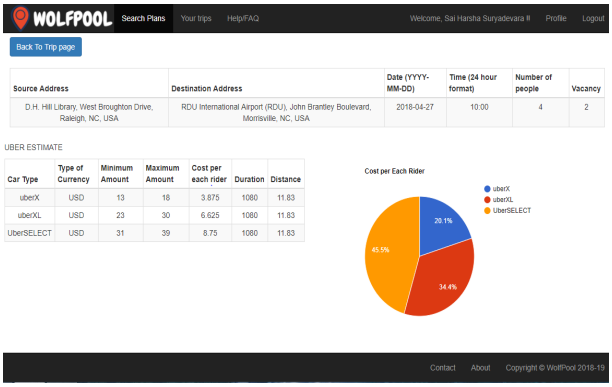

Figure 4: Trip cost Estimation

6.5 Edit or Delete a trip

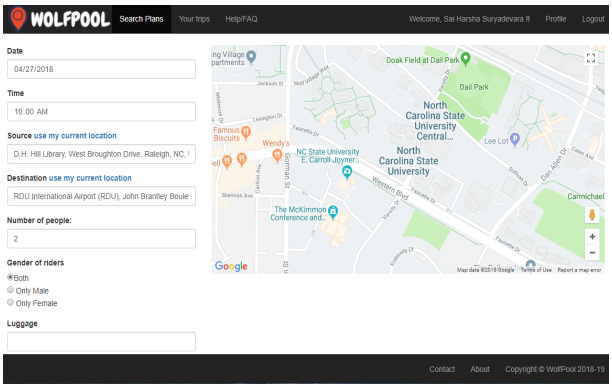

Figure 5: Edit Trip

Editing and deleting a trip features are provided to the users. The user can always cancel a trip and let others know that the user will not be sharing the ride with the other users. Here, the user will be taken out of the trip but other users still share the trip and the trip will always appear in the search results if it is a good match. The complete trip will be deleted only when there is one user in the trip and that user decides to cancel the trip. Similarly, the user can also update the trip by changing any of the parameters like source address, destination address, date of travel, time of travel, number of riders on the trip. Once the user updates the current trip, the user will be taken out of that trip but the trip will still exist in the database and the other users will be part of the trip. For this user, a new search is done with the updated parameters and if there is a match to any of the existing trip the user can then join that trip or create a new trip.

6.6 Chat Room

A chat room is implemented in the application where the users in the same trip can talk about other details about the trip. Every existing trip will have a chat room where all the users in the trip can chat among themselves and let others know if someone is updating or leaving the trip or has any additional requirements like extra luggage or bringing a pet in the trip.

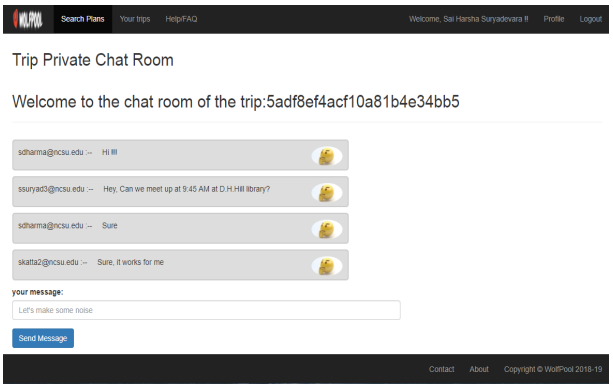

Figure 6: Chat Room

6.7 User Reviews and Ratings

The users can give reviews to other users who are part of the trip. The user can also give comments and feedback to other users regarding how well the trip went and how was it to share the trip with other users. These ratings and reviews will be used in the advanced search options where the user can search a trip expecting a minimum rating from the other riders on the trip. The trips with users having highest ratings will be displayed first and the trips with users having least rating will be displayed at the bottom which will help the users decide to join a particular trip if there are multiple matching trips available.

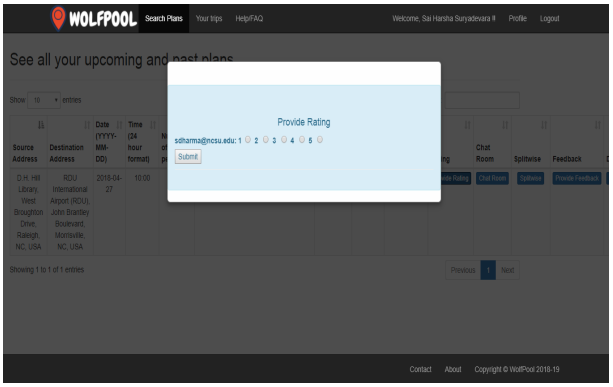

Figure 7: User Rating

7. EVALUATION PLAN

An evaluation plan is very important to test any application and to check if the application has all the requirements that were given before the development started. An evaluation plan is also important to find out any possible bugs and fix them. After the development phase was completed, a beta version of the application was given other people for them to test the few features of the application are-

- The user must be able to update and delete the trips. The updated/deleted trips should still be present in the database if there are other users in the trip. The trip should only be deleted if the last user is dropping out from the trip.
- The user should be able to get price estimates for the selected trips from Uber and Lyft. The user should get details about all the three different cabs provided by both Uber and Lyft. The details like minimum price, maximum price, duration, distance and average price per rider in the trip should be displayed correctly.
- The user must get the best match for the trip based on the advanced search options like gender preferences, luggage and minimum rating from other users. The user should also get other matches in the decreasing order of a number of preferences matched.
- The users in the same trip can chat with other users on the trip. The messages should be reflected immediately in the chat room. Once the user is out of the trip, the user should not be able to chat in that room. The user can join multiple trips and hence can chat with multiple users in each trip.
- The user should be able to rate other users who shared the trip with the user. The user can also give comments to other users. This rating should be used for additional search options. The users can join the trip only if the other users have a minimum rating.
- The user must be able to track the expenses for a trip and be able to share the cost of the ride. Any user can add the amount paid by that user into the expense tracking feature and this transaction should be updated in the database and all the other users in that trip owe the user who paid the cab bill. This should be reflected in each user's account.

Quality Measures:

- Security: The application will provide a secure login of the user and will send an email to confirm the email.
- Concurrency: Load balancing is one of the important quality measures. The application can support multiple users at once.
- Availability: The application will always be running and the servers will always be up.
- Adaptability: The application can be easily changed to other technologies and is highly adaptable.
- Scalability: The application can always handle increased user base.
- Performance: The application will be fast and would load each page without any delay.
- Reliability: The application will turn to the fallback server when needed.

Based on the evaluation plan, a survey was conducted and given to other people to evaluate the beta version of the application.

8. SURVEY

8.1 Survey Goals

The survey is an important source to get a real feedback from the intended users. A subjective approach might be intuitive in the initial stages, but it is highly recommended to move forward only with the objective information. Before launching a new product or application, we need to consider various factors such as acceptability, trust, willingness, and feasibility. Positive survey results reinforce the fundamentals of the product or design and provide some insights necessary to improvise to reach the target audience in an effective way. As WolfPool is a relatively new idea, it's very important to scrutinize the acceptance and feasibility of this product.

8.2 Survey Results

In this section, we shall briefly go through various questions posted in the survey and responses on the same. The survey was conducted on roughly 25 students from North Carolina State University.

8.2.1 Survey results

Below responses and questions are collected, with 21 people surveyed representing a normal user of this application.

Question 1.Did you find any bugs while evaluating the first version of Wolfpool? The link to first version is given below.

This question was aimed to find out if there are any bugs in the first version of the application and fix them as a part of enhancement. The users were given a text area to respond. Most of the users responded saying the search results were inaccurate and the search was returning trips that had different times. This bug was fixed in the application.

Question 2. Are you able to register and log into the application without any problems?

Figure 8: User Question 2

This question was aimed to find if the user is able to register into the application and if the user is able to log into the application properly. All the users were able to register and log into the application properly.

Question 3. Are the price estimates for a trip provided by Uber API and Lyft API accurate?

Figure 9: User Question 3

This question was aimed to find out if the fare estimates of Uber and Lyft were shown correctly and if the estimates were getting updated properly with increasing distance. All the users found that the estimates to be correct and accurate from Uber and Lfyt for different trips.The trips which were not supported by Uber and Lyft did not give any results.

Question 4. Which of the feedback options did you find appealing?

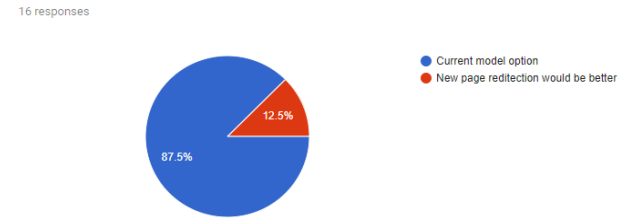

Figure 10: User Question 4

This question was asked to find out which feedback option did a user find more appealing. Around 90% of the users selected current model option. In this current model option when a user wants to give feedback to the fellow rides after the trip he can click a button which pops up a modal with text area field and where he can give the feedback. Around 10% of the users wanted the feedback text area in a new page rather than a modal.

Question 5.How good is the response time of application?

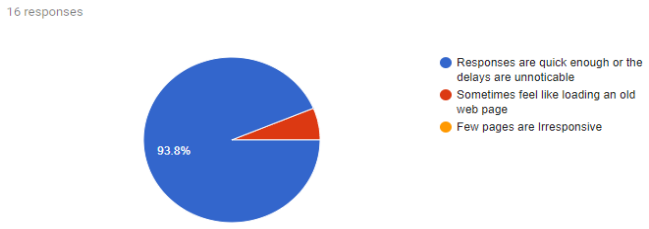

Figure 11: User Question 5

This question was asked to find out how good is the response time of the application to check how fast is data being retrieved and store in the database. Almost 90% of the results were stating that the responses are quick enough or the delay was unnoticeable. Around less than 10% people felt like loading an old web page. None of the responses were like pages are non respective.

Question 6. For multiple users in a trip, is the Edit and Delete feature updating the trip accurately?

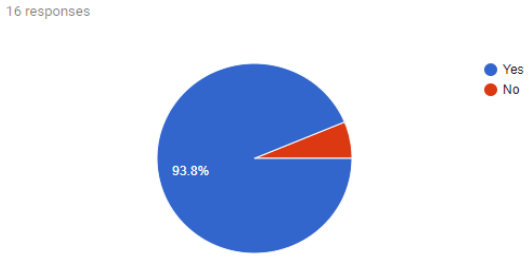

Figure 12: User Question 6

This question was aimed to find whether the edit/delete a trip functionality is working properly or if the users found any issues while updating or deleting the trip. Most of users were able to edit/delete the trip successfully which had multiple users in it. Less than 5 percent of the users had some problems with the edit option, but none of them had any issue with the delete feature.

Question 7. On a scale of 1 to 5, how would you rate the user experience of the wolfpool application?

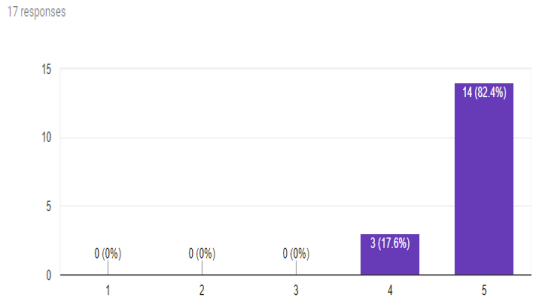

Figure 13: User Question 7

This question was aimed to find the user experience and the UI of the application for the different features that were added during the development in the second phase. Most of the users gave a 5 rating of the UI and few of them gave 4. Upon surveying, we understood that most of the users liked the simplicity of the application and the way it handled most of their requests accurately without compromising their experience. Most of the users were happy with the way the price estimation feature was implemented.

Question 8. Which was the most useful feature in the application?

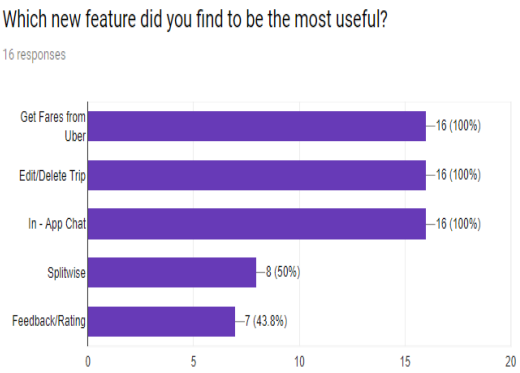

Figure 14: User Question 8

This question was aimed to find which was the most useful feature that was added. Most of the users found all the features useful, but most of the users found price estimate by Uber and Lyft feature the best which was followed by edit/delete feature followed by the additional search options.

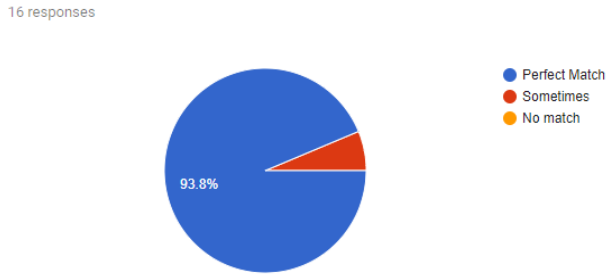

Figure 15: User Question 9

Question 9.How accurate were the search results for a similar trips?

This question was aimed to find if the advanced search results were accurate. Most of the users got the best-matched results to their search queries in the sorted order. Most of the users who were surveyed claimed that the search results were accurate in all the scenarios they tried.

Question 10.How fast were you receiving messages from other users in the chat room?

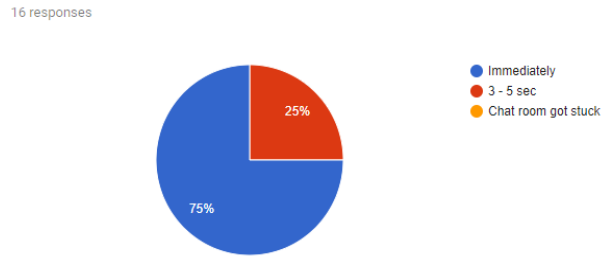

Figure 16: User Question 10

This question was aimed to find the response time of the messages received in the chat room of a particular trip. Most of the users received the messages immediately when multiple users messaged in the chat room. Only a few of the users received the messages between 3-5 seconds and only 1 user found the chat room to get stuck.

Question 11. Is the expense tracking option working as expected and are the changes reflected in other user accounts?

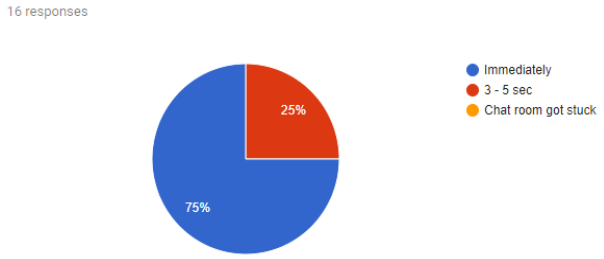

Figure 17: User Question 11

This question was aimed to find if the expense tracking option is working properly and if one user enters an amount as paid by the user, is it reflected in another user account proportionately. All the users tested and said that this feature is working properly.

Question 12. Was the built-in split wise application intuitive? ?

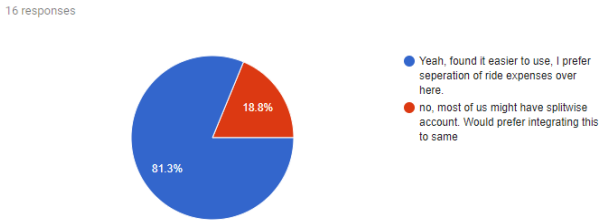

Figure 18: User Question 12

We asked this question to check whether the users liked the built-in split wise application or they would prefer integrating with the existing split wise application. Around 80% of the users were in favor of the built-in split wise application. But around 20% of the users wanted to collaborate the existing split wise application into the app.

Question 13. Do you want the expense tracking feature to be in the application or do you prefer integrating splits to the application?

This question was aimed to find if the users are satisfied with the expense tracking option or do they want the built-in splitwise application to be integrated into the application. Majority of the people wanted this feature to be part of the application as once the payment gateway is integrated, it would be great to send/receive money only for trips if it is

kept separate. But few people wanted the splits app to be integrated into the application.

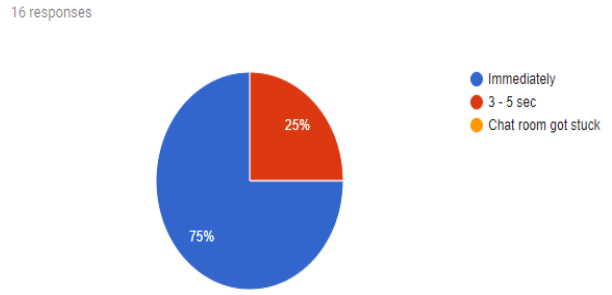

Figure 19: User Question 13

9. COMPARISON WITH FIRST VERSION

Before getting started with the coding part of the wolf pool application, our developers played around with the first version of the application. While doing this, we noted the bugs in the application along with the areas which required enhancements. After completing the debugging phase, our first phase of development revolved around fixing the high priority bugs. One major bug we found out was the search feature of matching rides. We created sample rides in future and we tried to check matching rides for the current date. Surprisingly, we saw that the software displays the matching rides in the future too. The software is intended to display close ride matches to the people in search of the similar ride plan. We realized that this has to be fixed immediately as the search feature is the most basic and important feature of the application.

It takes 20 years to build a reputation and five minutes to ruin it- said Warren Buffet. Even the best application with this kind of issues will not gain back the trust of the user, once lost. After hours of debugging, we found few relevant issues, necessary fixes. The issue related to the search feature was missing date constraint on the search results. The software captures exact match of the source and destination matching rides, it might be time constraint of the projects that might have related to missing date constraint. Few other issues were related to the search feature, delayed email feature, minor UI fixes in the first version of wolfpool.

The first version of the application was given to the users to find out if there are any bugs or if they could give any suggestions o improve the application, one of the most repeated suggestion was to remove the alerts that were displayed during the search or while creating a plan. The alerts were removed and a better UI was developed which gave more options to the users on how to respond when something fails.

When it comes to performance, the user will expect the matching trips to be displayed as soon as the user searches with some query. The search performance was improved by changing the database model and writing efficient queries and by filtering all the required records in the database query instead of fetching more records and filtering them in the server side application. The impact of this modification can be observed when there are a lot of trips that are matched with the search plan.

10. FUTURE WORK

Payment gateways like Paypal, Venmo, cash app, circle pay can be integrated into the application. If these payment gateways are integrated into the application, transfer of money will be very easy and the user can send/receive money from other users who share the trip. One more feature can be implemented which forces the user to have a minimum balance to join a trip or create a plan and the cost of each user in the trip will be transferred to the user who actually creates the plan and that user can pay the transport providers like Uber or Lyft. This will make the whole process so easy and transparent.

Another feature which can be implemented is charging a minimum penalty to the users who drop out of the plan at the last minute. Any threshold like 24-48 hours can be put as a limit to drop out of a plan without any charge. If the user drops within this period or if the user does not show up on the trip day, the user will be charged with some penalty. This feature is possible only if the payment gateway is implemented.

Once the above features are implemented and if Wolfpool becomes ready to use application, the application can be migrated to other platforms like Android. As some of the users may want to join the trip late and want to do it fast, a mobile app with a good user interface would be great.

11. CONCLUSION

Every person wants to travel comfortably but even economically. Wolfpool is an application which provides a platform where users can share rides with other users in the application who have same travel plans. One of the unique features of Wolfpool is that Wolfpool lets the users plan the trips well in advance with other users and plan multiple trips at once. The user can search for a trip with basic requirements like source, destination, time, date and number of people. The user can also add few parameters like gender preferences, luggage and minimum rating to the search option to get the trips which exactly match to the user's requirements.

The applications have a get estimate feature which lets the users get a rough estimate of how much the trip will cost. These estimates are reliable and accurate as the trip estimates are provided by Uber and Lyft. The application also has a chat room for each trip where the users in the trip can communicate with each other to resolve any conflicts. The application also has an expense tracking feature where the cost of each trip is shared among users and stored in the database and the user can always check how much the user owes or how much is owed. The users can also give ratings to the users with whom they share the ride and this rating will be useful for the users who expect a minimum rating from other users who share a trip.

These features are currently not being provided by leading transport service providers like Uber and left. Wolfpool is considered secure as the application only lets the students with .edu domain register in the application and share the

trip with others. It is also secure as the users can talk to other people in advance and share a ride with them, unlike uber pool where the user is forced to share a ride with a stranger. Wolfpool would be an ideal application for the users who would like to plan the rides in advance with known people at an economic cost.

12. ACKNOWLEDGEMENT

We would like to express our sincere gratitude towards our Professor Tim Menzies and our mentor Amritanshu Agrawal who have guided us throughout our project and helped us with their valuable suggestions. We would also like to thank our fellow peers who have given us their valuable insights from time to time which enabled us to develop this application in an efficient way.

13. REFERENCES

[1] Intelligent Carpooling System A Case Study for Metropolitan Area *http://tiny.cc/dhn8ry*

[2] Use the Maps API to show the real-time location of the customer *http://tiny.cc/t0o8ry*

[3] Car Poolup Real-time Carpooling using GPS *http://tiny.cc/u2o8ry*

[4] School of Science and Engineering Carpooling Application *http://tiny.cc/z6o8ry*

[5] MongoDB vs MySQL *https://www.mongodb.com/compare/mongodb-mysql.*

[6] AWS vs Heroku: Cloud Platform Comparison for 2017. *https://hackernoon.com/aws-vs-heroku-cloud-platform-comparison-for-2017-5f2194c0673e*

[7] Mean Stack. *https://www.brainvire.com/basic-guide-mean-stack.*

[8] Cloud platforms compared: Heroku, AWS, Azure, Digital Ocean, Google App Engine. *http://selbielabs.com/cloud-platforms-compared/*

[9] Developers j Uber. *https://developer.uber.com/docs/riders/references/api/*

[10] Lyft's API *https://developer.lyft.com/docs/overview*

[11] Express - Node.js web application framework. *https://expressjs.com/..*

[12] Heroku vs. Google App Engine vs. AWS Elastic Beanstalk. *https://stackshare.io/stackups/aws-elastic-beanstalk-vs-google-app-engine-vs-heroku*

[13] I. Lunden. Disrupt NY 2016. *https://techcrunch.com/2016/05/10/uber-says-that-20-of-its-rides-globally-are-now-on-uber-pool/*

Chits:
BXJ
FLI
FSD
LXK
WHP
OBP
OSD
EQO
BJA
PGC
HXU
PKV
AFK
UQJ
TLH
DNR
KIE