WolfTutor : A Peer-to-peer tutoring system based on reviews and rewards

Mateenrehan Shaikh
NC State University
Raleigh, North Carolina
mshaikh@ncsu.edu

Riken Shah
NC State University
Raleigh, North Carolina
rshah9@ncsu.edu

Himani
NC State University
Raleigh, North Carolina
hhimani@ncsu.edu

Aaroh Gala
NC State University
Raleigh, North Carolina
agala@ncsu.edu

ABSTRACT

We present Wolftutor, a peer to peer collaborative tutoring system that aims to supplement and enhance the learning experience among NC state students. According to our preliminary studies, the current proportion of students who have felt the need of quality tutoring is considerable. Even though students have an amazing faculty and teaching assistants at their disposal, sometimes students need more in depth knowledge and focused attention, which might be difficult for a teaching assistant or a professor to provide in a large class. Also, students admitted that they are more comfortable in asking silly doubts to their peers mostly because there is no fear of losing marks or making a bad impression or sometimes they are simply shy. The goal is to enhance the overall learning experience among students as it is said that "to teach is to learn twice".

Keywords

Peer Tutoring System, Tutor, Tutee, Students, University, Credibility, Predictability, Availability, Security, Adaptability, Incentive, Cost model, Slack-bot, Rewards-system.

1. INTRODUCTION

Tutoring is a learning experience in itself. However, the free tutoring services that are currently available, are not very effective and cannot cater to the personalized student requirements and those services which provide personalized training are not very affordable and accessible at ground level to students who need them. Hence, we came up with this idea of peer tutoring in order to fill this niche[5].

The peer to peer collaboration platform will be provided by us for use by students. Here the students themselves tutor other students in their respective areas of expertise. The general flow of the application would be like this. With signing up, they receive a few free reward credits which can be used to arrange initial tutoring. The reward credits can be translated to something tangible like discount coupons or cash. However, later he or she needs to earn more points by tutoring other students. After the tutoring session, the student can rate the tutor, and all such reviews are publicly available online along with other data like a number of tutoring sessions conducted by this tutor, etc. which would enable the student to validate the credibility of the tutor. Tutors can set up their price in terms of points per hour and can create their detailed profile indicating their expertise. New tutors may give some free tutoring sessions in order to build their profile and get good reviews from peers so that later they can earn points for their efforts. Both students and tutors (ideally both are students) have to agree to terms and conditions detailing the university policies and sign them indicating that they will not violate that[6]. The goals of our system are as follows:

1. Rewards based system which enhances knowledge sharing among NC state students.
2. Provide high quality tutoring through the platform that we build which will help students clear their concepts and succeed in the course.
3. Tutors will get experience of teaching and will increase their confidence and will enhance their subject skills.
4. A review and rating based system to maintain credibility of tutoring provided, student can see reviews and ratings prior to booking a tutor.
5. Tutors can share their success tips which can help student excel in that particular course.

We have created a Slack application that can be added to any Slack workspace. We have designed various user query flows with which the user can interact with the application. We have hosted our backend on NodeJS server and for frontend we are using various Slack components like dialog boxes, message menus and other interactive elements. Instead of using the traditional text based chat interface for user interaction, we figured, using interactive elements it becomes easier for user to communicate thoughts more effectively while having full control over the flow of the application.

The rest of the paper is divided as follows. Section 2 talks about the similar solutions adopted and how they relate to our solution. In section 3 we describe in detail, all the use-cases of our system along with illustrations to make it easy and clear to understand the application. In Section 4 we outline the underlying architecture of the system and describe the pipeline that we developed to automate various

processes. In section 5 we talk about various methodologies and how we adopted one. Section 6 talks about the evaluation we conducted and results of that evaluation on the basis of various non-functional requirements. Section 7 talks about the future scope of the project. And we conclude in Section 8.

2. LITERATURE REVIEW

We did a careful analysis of existing systems for peer to peer tutoring. Many universities have set up this system but we could not find a system which is based on peer reviews and rewards point system. By careful study, we found out that the common motivation of all these peers to peer tutoring systems is to benefit the students facing difficulty with coursework by utilizing the infinite potential of students with excellent track record[7].

While we do have normal lecture hours for all the courses along with the TA hours to help with the doubts and coursework, some may question the need for peer tutoring. The majority of universities strongly believe that peer tutoring helps strengthen the foundation of a course and students get to learn new approaches to overcome the complexity at hand. Having tried and tested the system, many universities have done as much as share success stories on the university website to promote peer tutoring. A survey conducted in session 2015-16 at Haverford college showed that 100% students found the peer to peer tutoring useful, if not very useful. A survey conducted at Brown University showed that peer tutoring helped students finish assignments a week early [2]. As Haverford university correctly stated, "Peer tutoring is not remedial, it is Supplemental".

Harvard University has a very interesting cost model, a graduate student who has gained expertise in a course can teach undergraduates in the same course, and in return gets paid 18$ per hour, where the undergrad student has to pay 7$ and rest of the cost, is incurred by the university [1]. Few of the systems charge a standard fee from student whereas in most of the fee is incurred by the university, the tutor gets paid nonetheless.

2.1 Motivation

Incentive Peer tutoring has been proven to be a win-win situation for both tutor and students, the incentive given to the tutor is often in terms of cash, or even points whereas the students benefit from the personal coaching experience. One very revolutionary approach that is often taken for peer tutoring is roles partway, where tutor and student exchange roles and often exchanging and explaining thoughts on a subject on what you learned is the best way examine if you have grasped properly or not. Another area where this can be helpful is when one student is good at maths, other can be good in English, these two can work together to understand difficult concepts. So instead of exchanging money or paying each other, we can establish this exchange of information.

2.2 Existing Systems

There is two common implementations of peer tutoring system. Online portal and Offline processing by college authorities[4]. The process starts by a student signing on the internal peer tutor portal or approaching concerned offline authorities and request for a tutor in the problem area or subject and university or portal will do the task of finding him one. In case of portals, some have implemented traditional scheduling algorithms whereas in the majority of the systems the requests or basically peer matching is done by concerned officials. Some universities like to go with more personalized approach and willing to be more involved to ensure the effectiveness of the system and they are in fact shelling money and resources to make this a success, whereas others have simply automated the parts of it, reducing resources and expenses by a little. Almost all the current peer tutor systems ensure the credibility of the system by restricting sign up to only enrolled students. Haverford university has an internal tool, authenticated by student login credentials where a student can go and search for a tutor. Apart from this a student also has to track his sessions with a brief summary of the tasks accomplished in the day.

There are some important ethical issues that do come into the picture, the student should never take help from the tutor for homework assignments and lab projects. To counter this, a proper orientation is given to students about the associated rules and regulations and how to ensure that they make the most of the program. There are proper channels for both tutor and students to contact the concerned authority person in case any of them is found guilty of violating any basic rules and regulations or simply because the student does not find the tutor helpful, so complete care has been taken for ensuring the usefulness of the tutoring program.

Different universities have combination of different modes of peer tutoring- there are drop in and appointment based where you can drop into a common facility at university and seek request and you will be served immediately or you can schedule an appointment as per availability, apart from this there is an online as well as offline mode [3]. Another important tutoring style could be single appointment vs weekly appointment. A single appointment is more like once you are stuck on a topic and you need help and the weekly appointment is more of a regular arrangement where a student needs helps with the coursework throughout the semester.

At North Carolina State University we have peer tutoring for undergraduate 100-200 level maths, physics and chemistry classes and we have seen its benefits and effectiveness of the students who take help from this tutoring centers. Peer tutoring at NC state if of two types- Drop in and Appointment based, The tutor gets paid by the university and the student does not have to bear any cost.

3. SYSTEM USE-CASES

In this section, we discuss some of the key use-cases of our application.

3.1 Sign Up

When a user starts conversation with our bot, he will type 'Hi' and he will get a message 'Do you want to enroll in Wolftutor', if he selects yes, we will get his basic information from slack using the slack user information api. We will collect the name, email and phone number of that user if

it exists. We then store that information in our database. once the information is stored in the database we will return the list of all the activities that a user can ask a bot. If he selects no we will just accept his response and will tell him he can enroll anytime to our system.

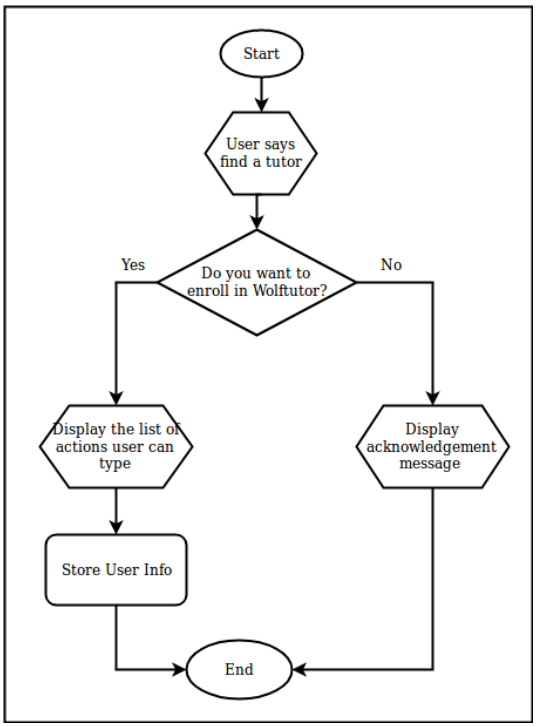


Figure 1: SignUp Flow

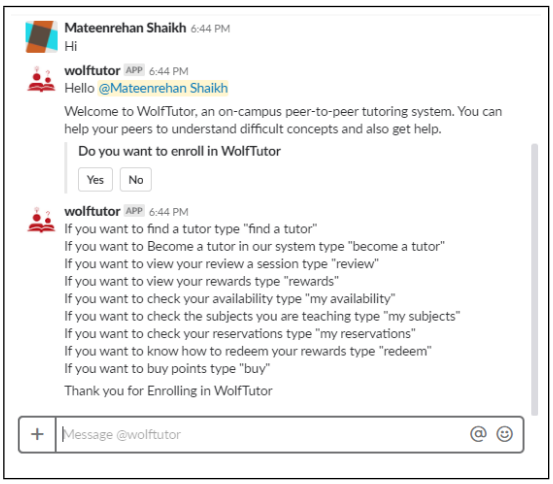


Figure 2: SignUp UI

3.2 Find a tutor

To search for a tutor the student will type find a tutor and he will be given a list of subjects from which he can select the subject in which he has difficulty. After he selects the subject, the student can see the list of tutors who are teaching that subject with their information like degree, the rate per hour, major, summary, and review of the tutor. Under each tutor, there are two buttons namely: review and schedule. By clicking the review button the student can see all the reviews of that tutor and by clicking on the schedule button he will be able to schedule that tutor which will be discussed in-depth in the next section.

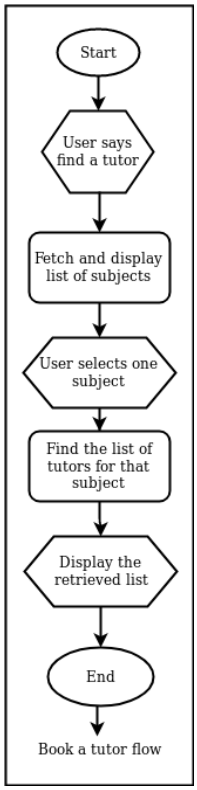


Figure 3: Find a tutor Flow

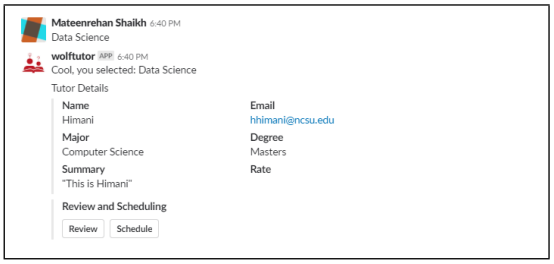


Figure 4: Find a Tutor UI

3.3 Book a tutor

Once the student gets a list of tutors he can either see the reviews and then schedule a tutor or he can directly schedule a tutor when he sees the list. Once the student clicks on the schedule button he will get the list of days during which the tutor is available. Now the student will select the day on which he is available and then our algorithm will give him the time-slots of 30 minutes on that day. This time-slot will be taken when the tutor fills his availability. If the students want a session for an hour then he can book two consecutive slots if they are available. Once the booking has been confirmed the tutor will get a direct message on

slack and an email notification will be sent to the tutor. The student can see his reservations any time by typing 'My reservation' command.

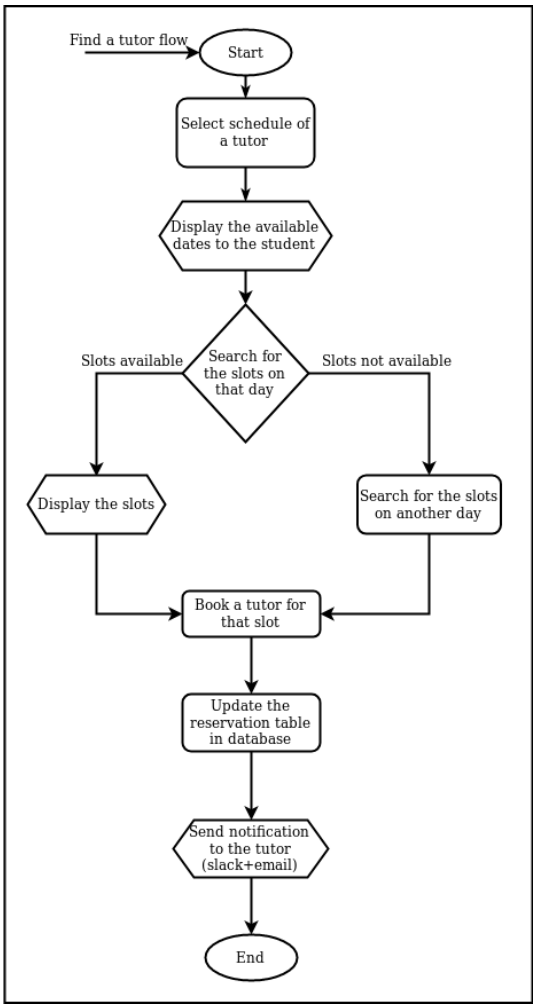


Figure 5: Book a Tutor Flow

3.4 Become a tutor

If a student wants to teach some subjects, then they will have to register as a tutor. If they type 'Become a tutor' on the slack bot then a very interactive form will open in slack where the tutor needs to enter his major, degree which he is currently pursuing, a rate that he wants to charge from students, a summary of his interests and why he wants to teach that subject. We have also given an option of keeping the rate 0 if the tutor wants to teach for free because in our initial survey we found that 28% students wanted to tutor for free. The tutors can update their rates anytime by checking the reviews and rating they get. Initially, we only take one subject from the tutor that he wants to teach and then we ask the tutor if he/she wants to add more subjects. Once the subjects are added we ask the tutors to add the availability i.e. the time that the tutor wants to teach per day. After all the information is added by the tutor we store the information in the database.

3.5 Reward and Review the Tutor

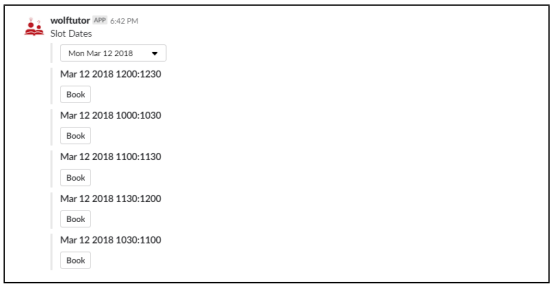


Figure 6: Book a tutor UI

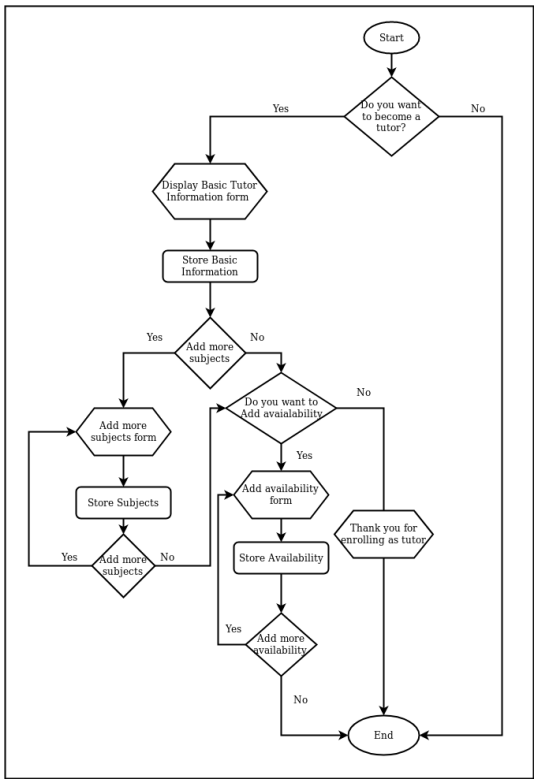


Figure 7: Become a Tutor Flow

On sign up, each user will receive 100 points which they can use to get the tutor. To learn or get help from tutor, the student needs to spend points for that or he can get a free tutor. The tutor will earn the points that he has set for each session with the student. The tutor can also covert these points into gift vouchers of university-based stores (For example: wolf-outfitters, various dining food courts). Based on the rating and reviews, a tutor can increase or decrease points for a tutoring session. If the student has no points he/she can even buy points from us.

At the end of a tutoring session, the student needs to rate and give a review to the tutor. This rating and review will help other students in searching a good tutor for help and it will also help the tutors in adjusting their points that they will charge in each session. It is the tutor's responsibility to ask the student to give him review after the session is over. To reduce the complexity of the project the student has to take the review by the end of that day or else he won't be

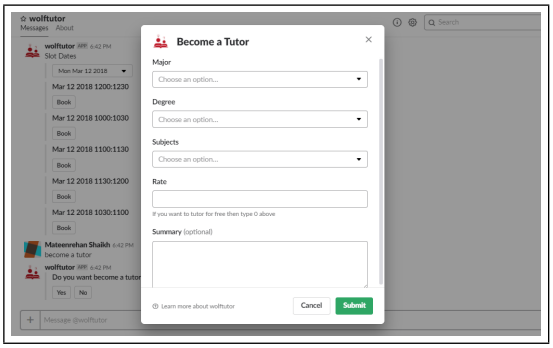


Figure 8: Become a tutor UI

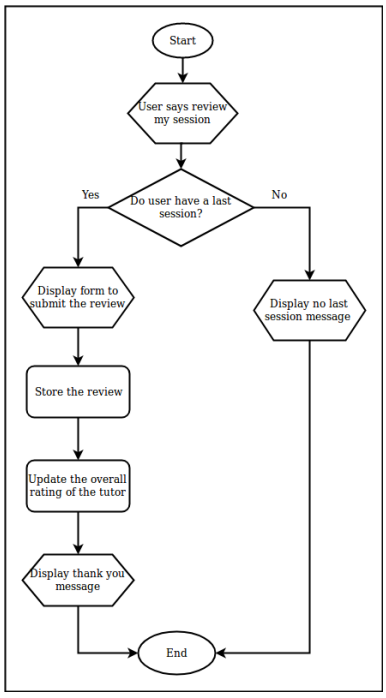


Figure 9: Review Flow

able to review that tutor.

3.6 Other Use-Cases

Apart from the above mentioned use-cases we even have some other smaller use-cases:

1. If a student wants to find his reservation he can simply type 'My reservation' and he will get the list of all his active reservations.

2. If a student or a tutor wants to check his rewards, they can ask the bot rewards and it will display their rewards.

3. If the tutor wants to check his availability, he can type my availability and he will be able to see his all his teaching availability.

4. If the tutor wants to check the subjects that he is teaching, then he can type my subjects and he will get the list of subjects he is teaching.

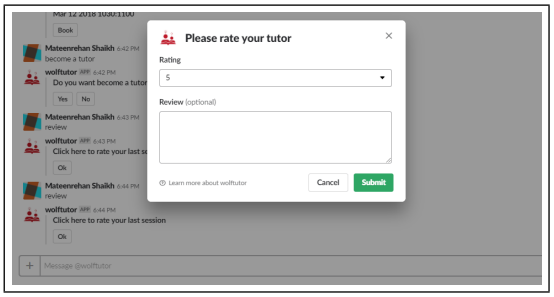


Figure 10: Review UI

4. ARCHITECTURE

The architecture of our system outlines the various components of the system and how they interact with one another. The detailed architecture is described in Figure 11.

Broadly speaking, we can divide the architecture in three main components. The slack app, the heroku cloud and the mongoDB database. This separation of concerns is a major advantage since database is independently hosted on mlab server and can be accessed from anywhere with valid authorization credentials. The heroku cloud hosts the logic of the slack app, which communicates with the user. We have also implemented the continuous deployment, continuous integration pipeline with Travis CI, which directly pushes the code to heroku cloud once the build passes (also indicated in the readme section of the repository). We have written test cases which acts as a sanity check for any commit made to the master branch, before deployment so as to not push broken code on the production server at heroku. Following is the detailed description of each component of our architecture.

4.1 Components of Architecture

4.1.1 Code Base

Code base is our Github repository where we maintain our code. Following the general convention, we made new branch for every new logical feature and also linked issues with particular merge commits. All this activity can be viewed in our repository. We have also used conventional practises of separating database queries and put them all together in *model* directory. Also all of our interactive components are in distributed in different folders.

4.1.2 Continuous Integration Module

The master branch of our repository is linked with Travis CI to be pushed to heroku server if the build on travis CI passes. The tests written in mocha acts as a final sanity check before deploying the code on the production server. Any code that is pushed on the master is directly built on Travis CI and deployed on heroku server if build passes.

4.1.3 Heroku Server

Heroku server is the home of our NodeJS application. We have used single dyno (free version) to host the application. The code pushed on master is deployed here by Travis and we always have the latest code in the production server. Also we have included the Procfile where it is indicated what command to execute to run the application (npm start).

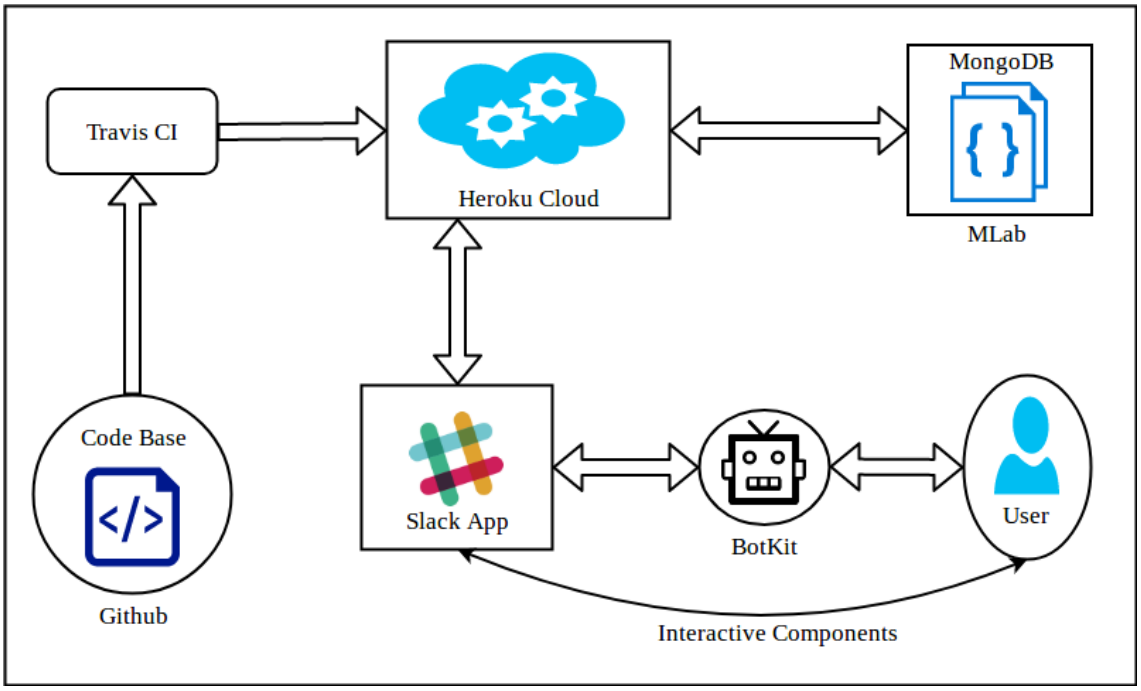


Figure 11: Architecture of the project

This is necessary for heroku to understand the starting point of the application.

4.1.4 *Database*

Mlab is a server that hosts our Mongo Database. The advantage of having a remote server for mlab is that everyone can access the same data at simultaneously. In mongo, only a mongo URI is required for accessing the database along with valid user credentials. This makes it very simple to test the application locally as well as run in on the server.

4.1.5 *Slack App*

Slack App is generated in api.slack.com dashboard. It is currently developed only for one workspace and it is configured to communicate to the heroku server which is hosted at wolftutor.herokuapp.com url. Also creating a slack app gives us various authentication tokens like access token and verification token which are required for Slack to authorize that the requests and responses are coming from a valid production server. Slack app also gives us configuration of the bot like its name its icon, etc. This will be visible to the user when the user interacts with the bot. Slack bot is a part of the slack app, where bot is considered like a user (bot-user) in slack terminology. Slack app also allows access to interactive components like dialogs and message menus for the overall user experience to be more rich and interesting.

4.1.6 *BotKit*

Botkit is an external library that integrates with Realtime API which can detect patterns in the user queries. The logic to be executed when a particular pattern is detected is written in the slack app (NodeJS code hosted on heroku). For instance, on saying *hi* user should be given an option to enroll in the system. This is one example of working of the Botkit module.

4.1.7 *User*

User is anyone who is signed in into the Slack workspace where the slack app is added. He/She can communicate with the app in variety of ways as described in the use-cases section. Also he/she can interact with the app using dialogs and message drop-downs and buttons to give a particular command. See details in use-cases section.

4.2 Bot Architecture

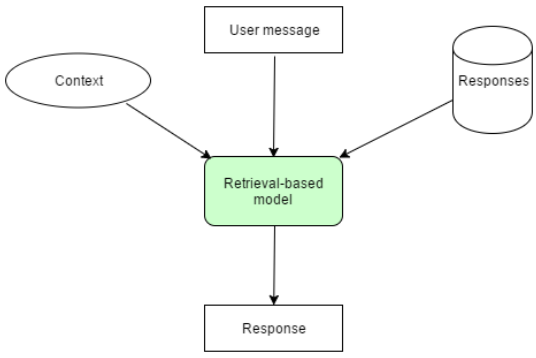


Figure 12: Bot Architecture

The Slack-bot behaves like a normal slack user and it can communicate with the actual slack user (and the user of the application) in the same way. Once the Slack app is added into the Slack work-space, users can straight away start communicating with it. The slack bot is based on a Retrieval based model where, when the user enters a query, the context of the conversation is considered as well. The context helps in deciding which path to go in the flow diagram as described in section 3.

5. METHODOLOGIES

5.1 Possibilities

We have found four different methodologies for solving the peer to peer tutoring problem. The first approach is the traditional approach in which there is a tutoring center where a tutor is available and the students can go there and he can get help. If the tutor is not available then an official can schedule the student when the tutor is available. The second methodology is a scheduling mechanism where the tutor and students are registered to our platform and the students can book the tutor and they can schedule the time and place to meet. The third methodology is an on-demand mechanism where the students can request the topic for which the help is needed. The tutors will see the requests and will try to serve the requests. If someone has requested a similar video before then the students can watch that video immediately. The fourth methodology is a real-time mechanism in which we will have an online platform where the students can get help in real-time. In this methodology, the students can select the tutor who is available and a real-time session will be created between the tutor and the students. We have implemented on-demand methodology.

Table 1: Evaluation of Different Methodology

Methodology	1	2	3	4	5	6
Traditional	-	+	+	++	-	- -
Scheduled	++	++	-	+	+	+
On-Demand	-	- -	++	- -	- -	++
Real-Time	-	++	++	+	++	- -

5.2 SlackBot Vs Facebook Bot Vs Web Application

We had an option of selecting slackbot, facebook bot or Web Application for implementing our project. We selected slackbot for several reasons:

1. Slack bot provides a feature called dialogs which act as a form which can take user response and would really enhance the user experience whereas facebook bot does not provide the feature to take information in a form. As we had to grab a lot of information from the user taking it in text format would be cumbersome for the user.

2. The web application would provide us with a good form but it won't be responsive i.e we will have to make different applications for different devices.

3. The slackbot can also be easily distributed and anyone can include the bot in their workspace and all the users in that workspace can use that application.

4. The slackbot also provide very easy revocability feature compared to the facebook bot.

5. If it would have been a social or commercial application we would have chosen facebook bot but as we have an educational application we selected slackbot.

6. The web application could have displayed the scheduling page better but due to its limitation of distribution, integration, security, and maintenance we selected slackbot over web application as well as facebook bot.

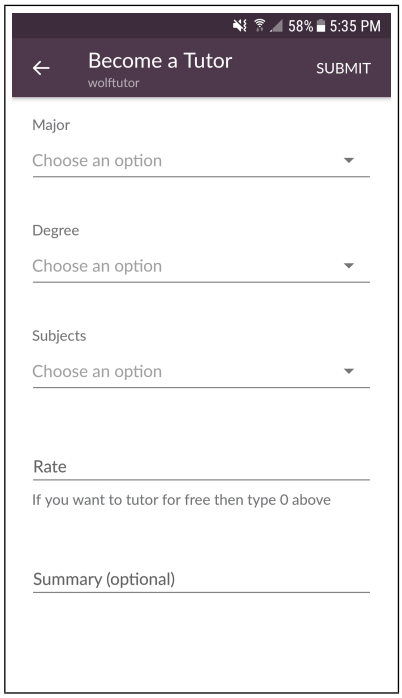


Figure 13: Mobile View of WolfTutor

5.3 Development Process

We have followed agile methodologies of development and have used kanban boards to track our progress. Here are the images justifying that.

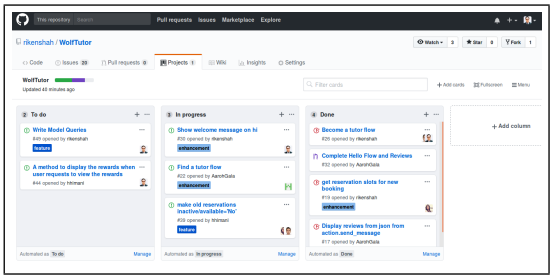


Figure 14: Agile - Initial State

6. EVALUATION

We had planned to test the effectiveness of our peer tutoring system by performing real-time evaluation among students. So, we booked a room in the D.H. Hill library to carry out the in-person evaluation. As per the groundwork, we had deployed WolfTutor bot on Heroku web server and asked interested students to fill in the evaluation sheet and choose their slot of preference for attending the evaluation. On the day of evaluation, once the students were done playing with the bot, we asked them to fill a short survey to rate the application across several parameters-Predictability, Availability,

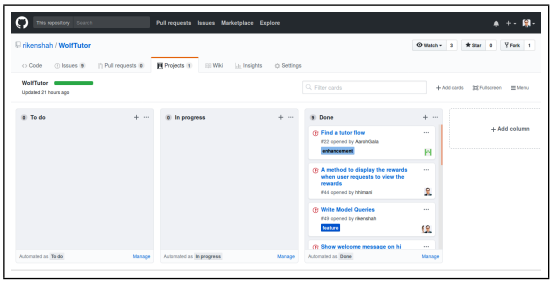


Figure 15: Agile - Final State

and Adaptability. We received an amazing and overwhelming response from 20 evaluators. Students in general really liked our application and complemented us on how smoothly everything was working.

Our goal was to cover the all the parameters discussed in the Evaluation analysis and more. We were able to cover Predictability Availability, Adaptability and Cost in the survey questions. Apart from these parameters we also covered Likeability, Future scope, and Usability. We have also explained a little bit in detail about platform compatibility and maintenance.

As per security, the system is already secure because we are allowing enrollment to only NC State students, also we are sending invites to access our wolftutor slack work space. We are using slack bot and slack APIs to achieve the functionality of peer tutoring application, so **Security** is Defacto provided by slack.

Speed can be cross referenced by high Availability of the system. We have used Node.js for the back-end development of slack bot, which is an event driven language, highly scalable and responsive and it can support a lot of concurrent users, therby ensuring high speed and availability at all times. To sum it up broadly, Evaluators rated application

Shows survey results of overall appeal of the idea.

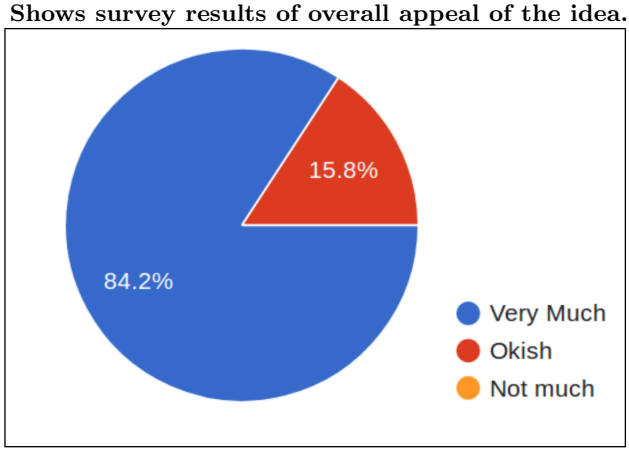


Figure 16: Likeability

as 100% in terms of **Likeability and Availability**.

As per **Cost**, Figure 17 shows that more than 75% evaluators loved the rewards system. whereas 25% really liked it. WolfTutor has implemented a sophisticated rewards based

Rate rewards system on a scale from 1-5.

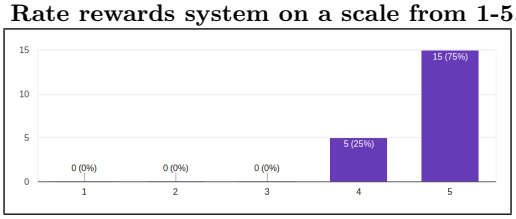


Figure 17: Cost-Reward system

system where Tutee pays in terms of points for availing tutoring services and the tutor gets paid in terms of points as well. Eventually we have given a policy for redeeming as well as buying points. Initially, the system is self-sustaining because 100 points are given to users on signing up into the system, which basically gives user a headstart.

Willingness to use the application in future

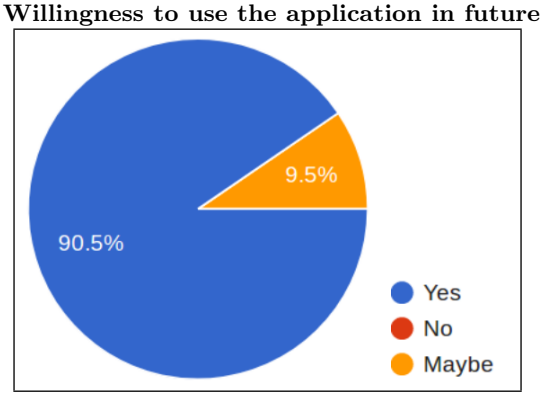


Figure 18: Predictability

As per **Predictability**, 81% evaluators said that they found the application flow intuitive. Also, 90% said that they would like to use the application in future.

Application flow being intuitive or not

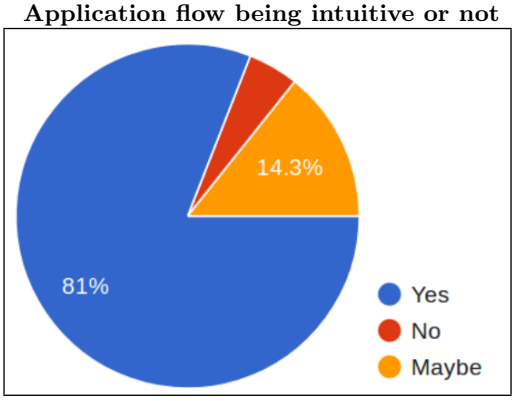


Figure 19: Predictability-Application flow intuitive

And 100% evaluators said that they founded the application as **Adaptable** i.e. They felt that the WolfTutor bot met all major expectations of a peer to peer tutoring system. Also users already accustomed with slack bot or chat bot functionalities in general found it really easy to use.

Around 85% evaluators rated our system as useful a.k.a **Usability**, where 100% found it easier to become to become

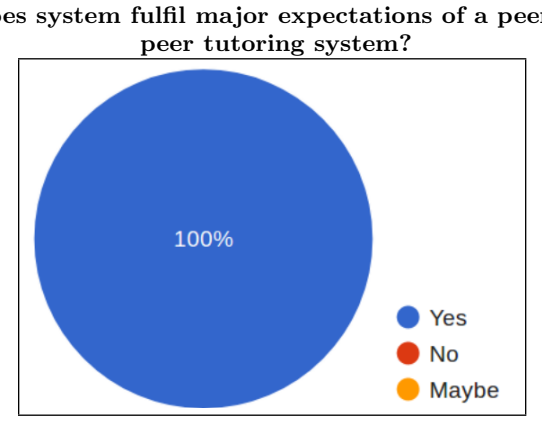


Figure 20: Adaptability

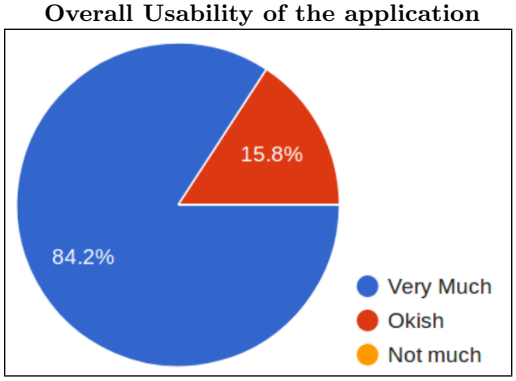


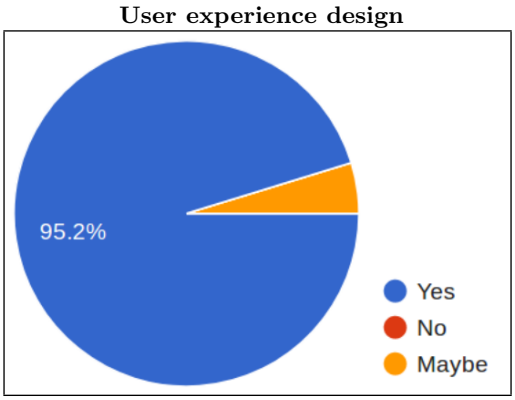
Figure 21: Usability

a tutor and 95% found it easier to search a tutor, which are the two major use cases of our application.

We would also like to discuss some other non-functional requirements met by our system, as per the **platform compatibility,**, our slack bot workspace can be integrated into web, mobile or desktop version of slack application hence making it accessible across all three platforms. This also makes our slack bot highly available. And if we discuss Maintenance of the bot, there can be some issue in the future when slack updates its API and does not support backward compatibility. But other than that, we are good in terms of maintenance.

As per some general overview of the application, 100% felt that our Readme was thorough and very useful in getting them started with the system. 95% evaluators liked our User experience whereas a whopping 100% enjoyed working with our system.

We also had an optional column where evaluators could report any bugs that they found in the system, there were no major bugs reported, our application was highly available and consistent throughtout the evaluation period. One crucial bug that was found, even though this is not breaking the flow but rate of tutors were not displayed when a student tries to book a tutor. Also, a few suggested pagination for display list of items, which could be list of subjects or tutors but this is limited by slack interactive messages API,



we have no such provision to show paginated results, hence we had to show scroll-able results.

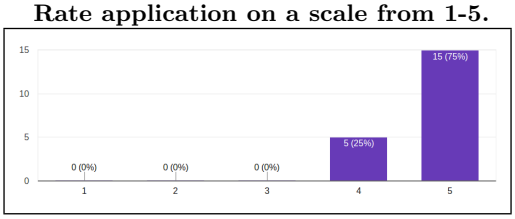


Figure 23: Usability-System level

And as per improvements, We should have to make bot smarter i.e. respond to versatile or smarter commands into the bot. This could have been done using regex when it comes to hearing commands from users or some sort of NLP. Another improvement or suggestion was that we should extend it to other schools and universities and make it a freelance tutor application and not limit it to just Wolfpack. Finally based on the survey results and exhaustive evaluation analysis, we have established overall effectiveness of Wolftutor.

7. FUTURE SCOPE

7.1 Online platform

Wolftutor is a feature rich application, the next goal in the future scope would be to give students a platform for conducting tutoring sessions online as well. As of now students can schedule an appointment and are given notifications for the same but they have to decide the place of meeting and tutoring session is held offline.

7.2 Scheduling algorithm

Our scheduling algorithm does given notification but a better idea would be to give both student and tutor an option to sync the scheduled reservation with their Google calendar. Also we have implemented a very basic scheduling algorithm, CRUD functionality is not supported. A user can only create and view his reservations, he cannot delte or edit it.

7.3 Notifications

As of now notification is sent to the tutor in case a tutee reserves him for a slot but no notification is sent to the tutee. Notification system can be further enhanced to notify users about their point updates.

8. CONCLUSION

As established before, Peer tutoring is not remedial but it is supplemental. Multiple research suggests that peer tutoring is very effective and sometimes it is twice as powerful as learning. Multiple universities have their own university peer tutoring program. NC State has a peer tutoring system which is very limited and an offline process. Also, WolfTutor is designed keeping university and students in mind for security and safety concerns.

Based on evaluation results and testing, we have established that our peer evaluation system works. Our four major use cases are fulfilling the expectations of a typical peer tutoring system. It offers high availability and platform compatibility. Also, as per our literature review and analysis, WolfTutor is the one stop online platform for all tutoring needs, we do have some sort of automation in existing systems but not one system provides automation of all the features. Also, the rewards based system encourages students to come forward to sign up for WolfTutor, whereas the review system maintains the credibility of the system. Our system reported no bugs during the evaluation period, only some basic tweaks and enhancements. The Slack chat bot is highly interactive and fun to use. We have successfully leveraged interactive messaging APIs of slack to make overall application appealing.

9. REFERENCES

[1] Harvard: Peer tutoring system. https://bsc.harvard.edu/pages/peer-tutoring, 2018. [Online; accessed 31-January-2018].

[2] Haverford: Peer Tutoring System. https://www.haverford.edu/academic-resources/peer-tutoring, 2018. [Online; accessed 31-January-2018].

[3] Penn-state: Reasons for peer tutoring. http://tutorials.istudy.psu.edu/peertutoring, 2018. [Online; accessed 31-January-2018].

[4] M. J. Evans and J. S. Moore. Peer tutoring with the aid of the internet. *British Journal of Educational Technology*, 44, January 2013.

[5] M. M. Kim. Peer tutoring at colleges and universities. *College and University*, 90, July 2015.

[6] T. Pugatch and N. Wilson. Nudging study habits: A field experiment on peer tutoring in higher education. *Economics of Education Review*, 62, February 2018.

[7] K. J. Topping. The effectiveness of peer tutoring in further and higher education: A typology and review of the literature. *Springer*, 32(3):321–345, October 1996.