Final level of achievement:

Gemini

Github:

https://github.com/TCK1997/Context

Purpose:

To create an app that gives users the ability to seamlessly open a website address on their laptop from a phone

Target Audience:

Mainly for students and working professionals

Initial plan for the project:

The application would provide an interface for users to share their link to. The user will login to the application on their laptop and mobile phone to sync.

After users have successfully managed to sync their mobile app and their laptop, users can either manually open the application and type the particular website address that they wish to open in another device or directly share the website address from their browser to the app. Once the link is shared, the computer connected would open up the link by the users.

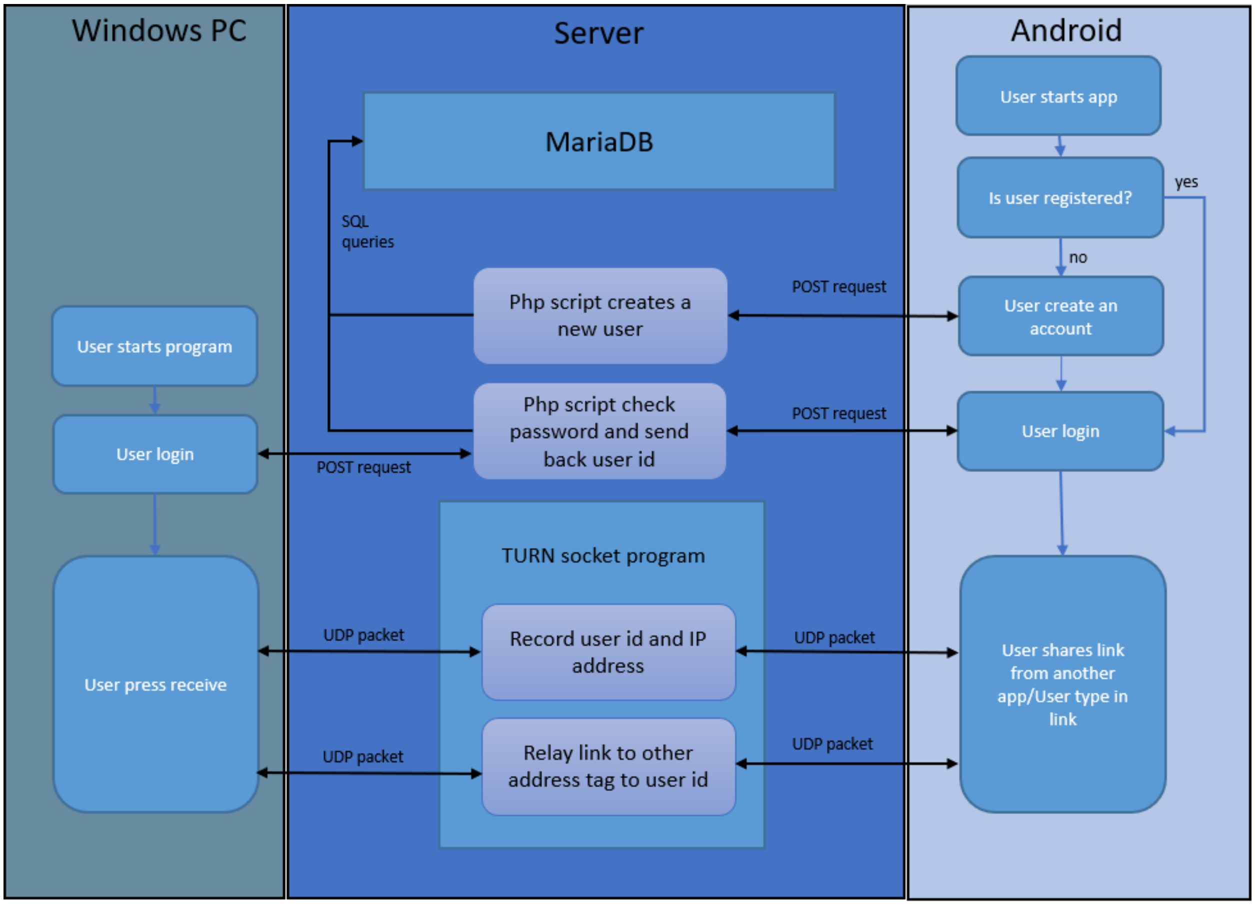
For this project, we wanted to abandon the usual client-server model and move towards a Peer-to-Peer (P2P) model. The initial plan was the use Session Traversal Utilities for NAT (STUN) and Traversal Using Relays around NAT (TURN) to circumvent the issues with NAT. However, STUN prove to be difficult as we could not do UDP hole punching through the symmetric NAT which are present in 4G network. We ended up relying on using TURN to relay most of our information. The end result is as good as a client-server program.

Overview of Total progress

For milestone 2, we have managed to create a basic app and winsock program that is capable of sending a website link from a user's mobile to the his/her user and open the link in their laptop browser. Both the mobile and the laptop connects to a turn server to find out the IP and respective port to start a peer to peer network between them and send data afterwards. In order to achieve this, we have learnt socket programming, coded simple client server networks and established and tried to design a peer to peer network.

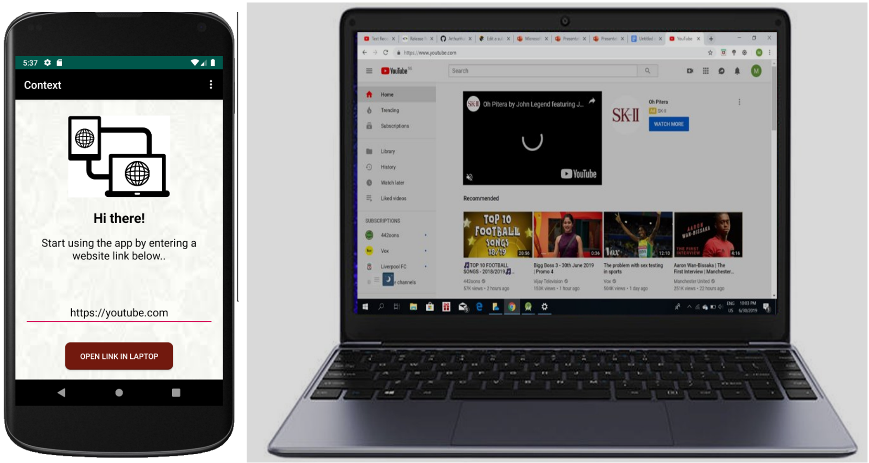
During milestone 3, we focused on coding the UI/UX of the android application as well as adding a login and register function so that we are able to differentiate between different users. We used web server to handle all the login and register request. For android, we used the default httpurlconnection inside the standard library to send the required request. For windows, we used cURL library to send the request. As the windows program was also lagging at the end of milestone 2, we threaded out some of the blocking socket functions to ensure that the windows graphics did not crash. For the UI/UX aspect of the android application, we took added introduction slides in the android app and beautify some parts of the application.

Main Program Flow:



Main Components:

1) Users have the ability to open a website link in their laptop through their phone. As seen below, after syncing their phone and laptop, users can manually type a website link in the app and open it in their laptop after pressing the send button in the mobile application.

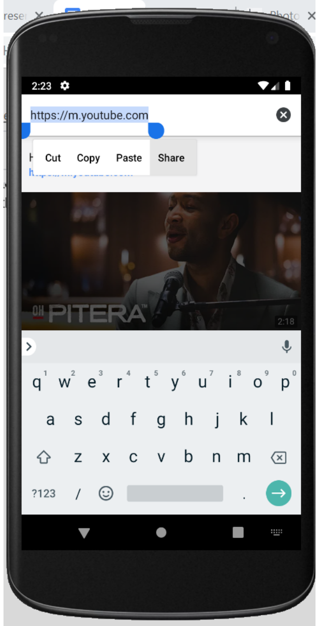


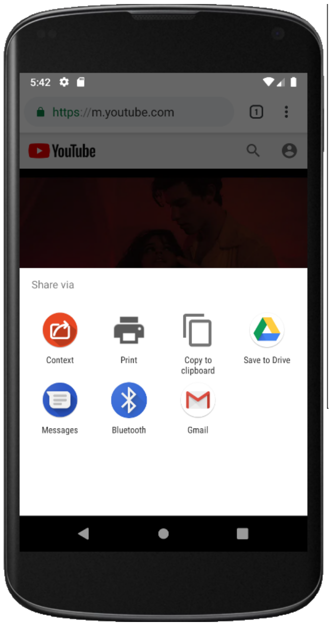
Hi there!

Start using the app by entering a website link below..

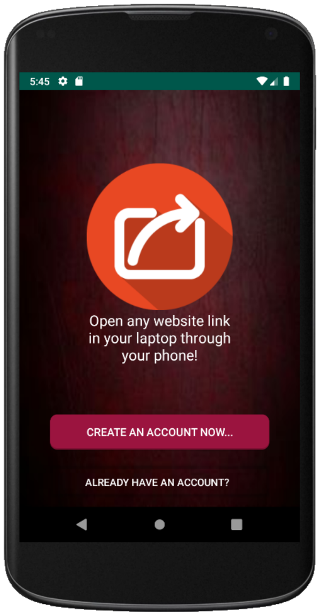
https://youtube.com

2) If the website URL is very long to type or if users are simply too lazy to manually open the app and type the link, they can simply share the link to the app. The shared website URL will automatically open in their respective laptop without the need for them to open the app.

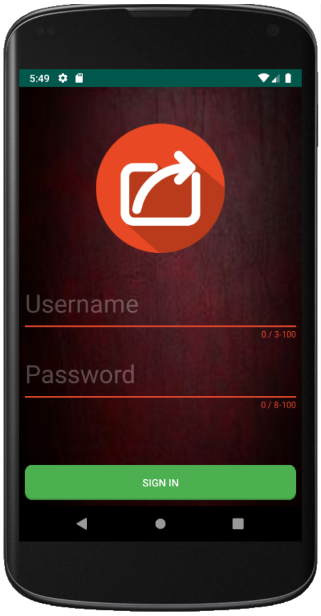




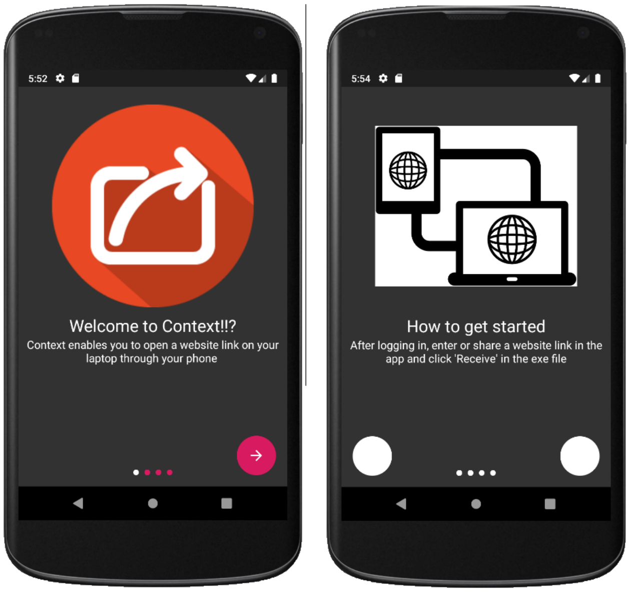
3) Users have to register/login before they can access the app. This makes it possible for more than 1 user to use the app at any one time.



Open any website link in your laptop through your phone!



4) We also have created intro pages which will be displayed when users open our app for the first time, These intro pages not only welcome our uses, but also provide them with information about our app and how to get started.



Welcome to Context!!?
Context enables you to open a website link on your laptop through your phone

How to get started
After logging in, enter or share a website link in the app and click 'Receive' in the exe file

Missed /Uncompleted features

- File Transfer
  We wanted to create a feature whereby users can also share files between their phones and laptops. We tried many methods and even followed many tutorials but were unsuccessful. We were indeed successful in transferring files through firebase. However, this method still required few modifications and as we did not have enough time, we had no choice but to drop the idea.

- Optical Character Recognition (OCR)
  We also wanted to create OCR inside our app but following feedback from our mentor that we did not have the need to have OCR inside our app, we decided to not do OCR in the end.

Problems encountered

1) We were very new to android studio, java language and socket programming as a whole. As a result, we did many errors and spent many hours debugging those errors.
2) We had difficulty developing the winsock program. We had a lot of difficulty developing the client and server systems. These include creating a socket for the client, connecting to a socket and sending and receiving data on the system.
3) After we were done with our app interface in android studio, we had difficulty integrating the app to communicate and respond with the winsock program.
4) It was quite difficult learning the win32 api. We think that we have made the mistake of

using win32 api instead of the Microsoft Foundation Class (MFC) library, or using C# instead of C++.

5) We had few issues about multi-threading and because of it there were times when our context app was hanging. The issue has been solved by threading out the problematic blocking socket function.
6) During milestone 2, only one user can use the services at any one time. We rectified this problem by adding in login and register function to differentiate between users.
7) We had issues with android UI/UX programming did not really understand the entire structure of the program. As a result, we spend quite a few hours debugging.
8) We wanted to package all the libraries together with the windows executable. However, we are unable to statically link the cURL and openssl libraries for some reason. We ended up zipping up the executable with the .dll libraries so that the users could still run the program.

Lessons Learnt

1) We could have learnt the concept of socket programming more thoroughly before we started coding. That way, we could have reduced some of our errors. We could have read more, watched more tutorials online and started with small basic programs before embarking on our main project. It would have made us efficiently spend our time and familiarise with what we are doing.
2) We need to have better communication between each other as there were times when we did not plan properly before hand and ended up doing extra work which have cost us a lot of time and effort. We could also have better delegated the job between us.
3) We could have utilised git more during the entire project in order to work on new experimental features without interfering with the main code of your project. We could also used git on the earlier stages of our project to organise our files and handle merge conflict.
4) In terms of skills, we have learnt to become better coders who are familiar with android and socket programming. Over the course of our work, we have get introduced to a wide range of libraries and coding hacks that we think will be definitely useful and helpful in our future modules and projects.

User Testing of our app

Between 26th and 28th of July, we conducted a small user testing of our app by asking our friends and family members to try our app and give us some feedback. The feedback was generally positive with many commenting that it was cool to open a website link in their laptops through their phones. But some of them indeed commented that there was a lot of scope for further and future upgrades and additional features like the ability to transfer files between two devices. As written earlier, file transfer was one of the features we wanted to add in but we were unable to due to time constraints and lack of prior knowledge regarding it. But we feel that this is not the end and maybe in the near future, with more knowledge and understanding of socket programming, we can revisit this project again to add in file transfer and other cool features. And another improvement that users wanted to see was better app design. Many felt that the app design was very plain and not very appealing. We have taken it seriously and have tried to enhance our app design and layout. We have also taken another step in improving our user experience by introducing introduction pages that will be displayed when users use our app for the first time. These introduction pages not only welcome our users but also give them information about the app and how to use it. This is evident from how our app currently looks compared to how our app was back in milestone 2.

\

Improvements for Milestone 3

1) Create an optical character recognition(OCR) for our app so that users can take a picture of a website link and access it. For this, we plan to use the google API library and android image cropper library.
2) Enable users to send website links vice versa from their laptop to their mobile phone (i.e open a website in their mobile phones from their laptop)
3) Create a unique ID for users when they login into our app.
4) Better design the aesthetics of the app interface to make it look unique and more attractive
5) If time allows, conduct user testing of our app to get their feedback and thoughts on our mobile application.
6) Create a product tour page when users open use our mobile app for the first time so that they are more familiarised on how to use our app.
7) Decide on the feasibility of having a file transfer option in our app (whether such a feature can be done by the end of milestone 3 and if so complete it).

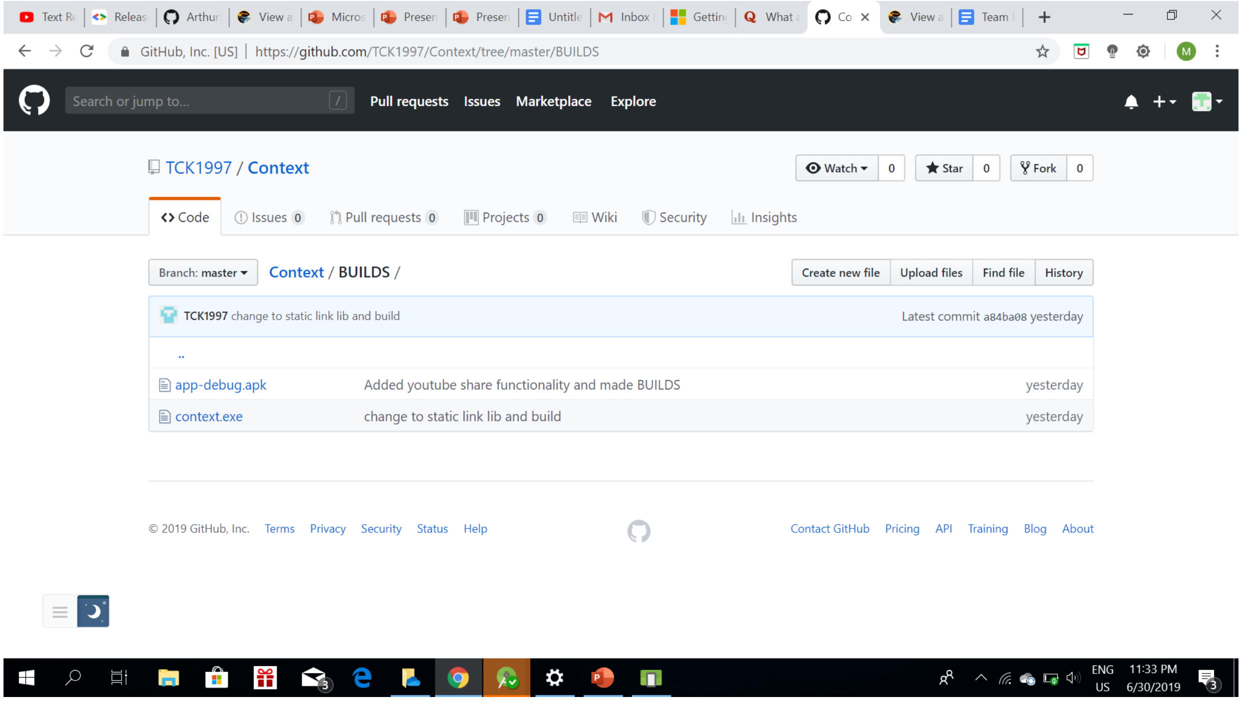
Execution Plan

What to do?	By:
- Finish up on evaluation on other groups milestone 3 - Work on other groups and mentor's advice and feedback - Work on splashdown - Ensure multiple people can simultaneously use the app at the same time without problems - Work on rectifying any other problems if any	By Mid of August

Final thoughts and reflections

These three months had been very meaningful and fruitful with the orbital project**.** During these three months, we have learnt a lot of things such as socket programming and android programming. We would like to thank all the profs, our mentor and other people who have given valuable feedback and advice to us regarding our project. To be honest, we are not so satisfied with our final product because there were a lot of other things that we wanted to do but could not despite our best efforts, time and dedication. Nonetheless, we feel that this is not the end and just a start for many other things/products we can build in the future. With respect to it, we do intend to revisit this project again to complete other features that we have not completed, once we feel that we have enough expertise and knowledge regarding them. Overall, we are extremely delighted to have finished orbital despite our hectic schedules and other commitments. We hope whatever we have learnt in orbital will help us in our future workplace and modules.

How to test and use our application?



GitHub, Inc. [US] | https://github.com/TCK1997/Context/tree/master/BUILDS

TCK1997 / Context

Branch: master ▾ Context / BUILDS /

TCK1997 change to static link lib and build — Latest commit a84ba08 yesterday

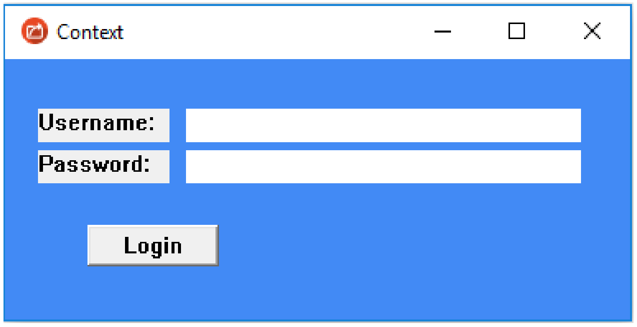
..		
app-debug.apk	Added youtube share functionality and made BUILDS	yesterday
context.exe	change to static link lib and build	yesterday

© 2019 GitHub, Inc.

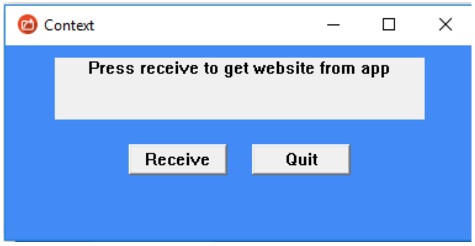
1) Go to https://github.com/TCK1997/Context. You can also view the current progress of the app and related files here. Click “BUILDS” and download the app apk file on your phone and the ‘context.exe’ on your laptop that you wish to open the website link on.

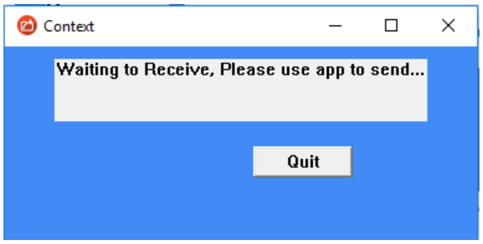
2) Install the app-debug.apk file on your phone.

3) Open up the application and register an account.



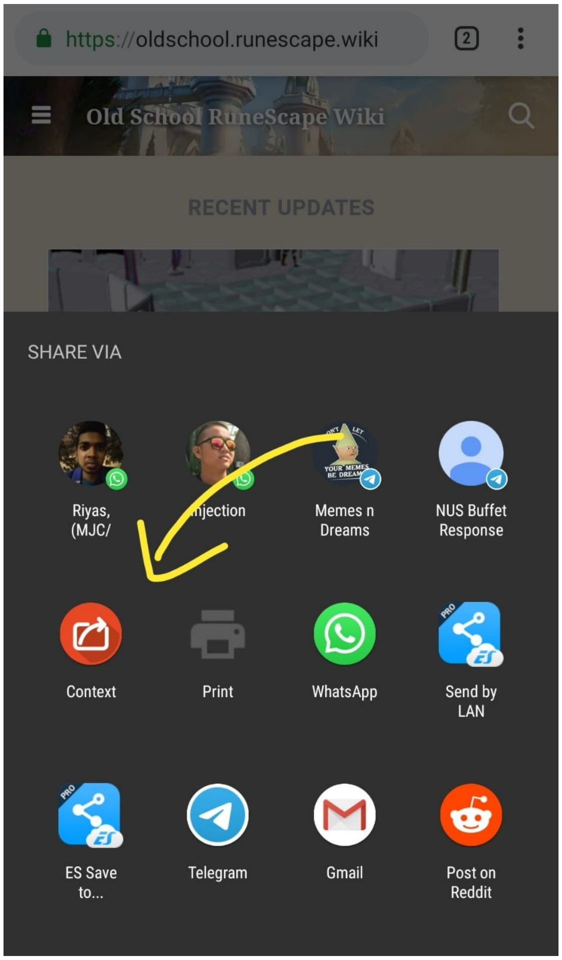
4) Once completed, open the ‘context.exe’ file in your laptop and you will come to this window. Login with the account you just registered on the application.


Press receive to get website from app


Waiting to Receive, Please use app to send...

5) Press receive on the windows program.

6) Now open the context app in your phone and simply type any website you want onto the text bar and press send



.

7) The app should now open up a website on your default web browser. The application would also work if you press share on google chrome or on the youtube application on your phone. Just remember to press receive on the windows application before sharing.