Project Title: TAPP
Team Goose
Team Members (Quest ID):

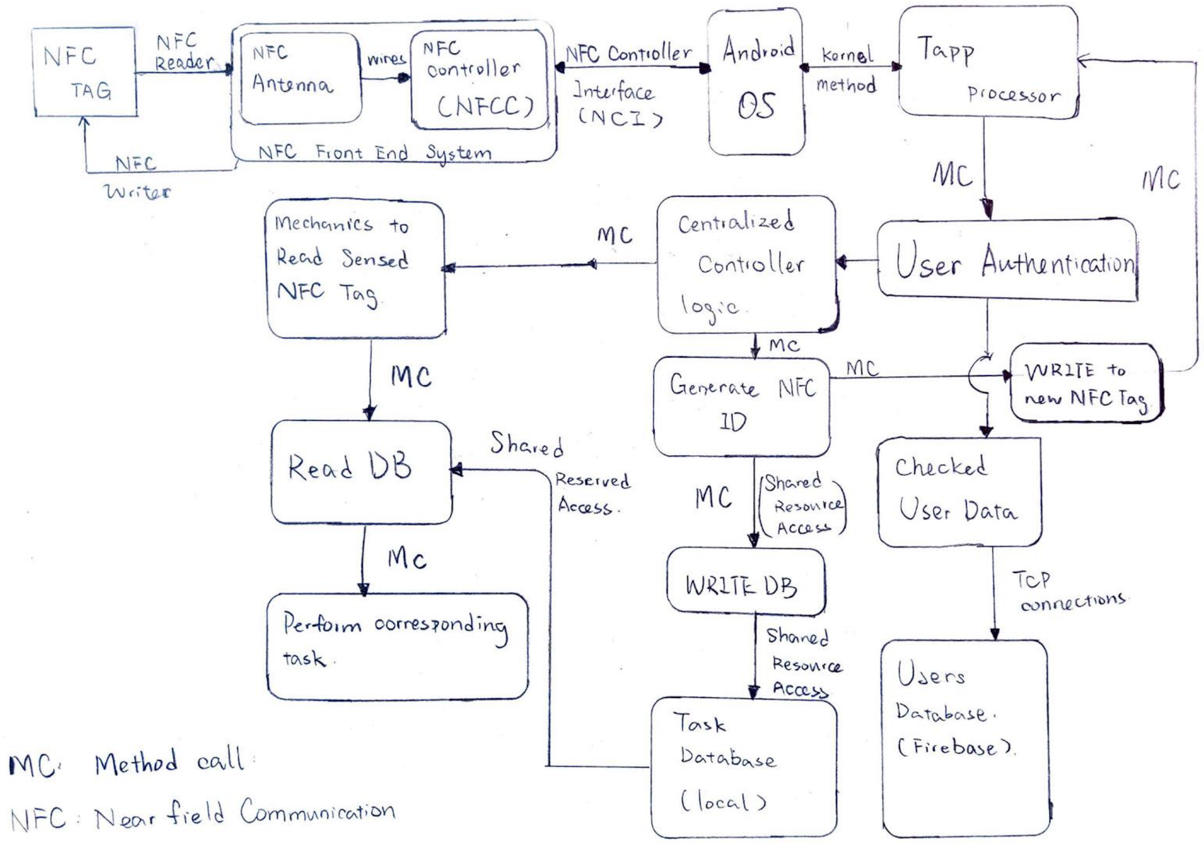
Anurag Joshi (a35joshi) Gunin Khanna (g3khanna)
Daniel Weisberg (dweisber) Xinjue Lu (x89lu)

We plan on demonstrating two functionalities during our demo. The first functionality is that the application will turn on the wifi for the Android device if it is not on already. The second functionality is that when a University of Waterloo student taps their phone on an NFC tag placed at the desired location such as their study table, the application will automatically open LEARN. The two functionalities allow the user to eliminate the overhead of turning on the Wi-Fi and accessing LEARN as the two tasks will be triggered through the NFC tag. The act of opening LEARN through an NFC tap is both real and simulated. It is real in the sense that we implemented a component that opens a web browser when the phone detects an NFC tap. Currently, we have hardcoded the string that directs the application to LEARN as we have not implemented the logic to handle addition of new NFC tags and configuring the respective options. Similarly, the user login page for user authentication exists, but it is a simulated action since the application isn’t connected to a user identification database yet. Therefore, the current workflow assumes a successful login regardless of what is entered in the text input.

The approach that we used for developing our system was a bottom-up approach. We tried to resolve any blockers or critical issues which prevented us from implementing any functionalities. The current progress on the system is that firstly, we managed to buy a whole lot of NFC tags/stickers for a very low cost in bulk. This solved our first problem which was the lack of enough NFC tags for each one of us to test independently instead of sharing the few ones which we had. Also, we managed to implement the logic to connect our phones to the NFC tags and get NFC IDs from the NFC tag to the phone and vice versa. Although this functionality exists through raw code, we still have to create the user interface to allow the end users to add and delete NFC tags and specify the functions that they intend to do with the application.

The major difficulty we are facing is that all the phones available in the library don’t support NFC and this makes coding harder for those who own non-Android phones. Another logistical difficulty we are facing is finding a common time during weekdays when we can all meet and discuss our progress since we are all swamped with interviews and other assignment deadlines. Over the next one month of development, we plan on trying to implement as many functionalities as possible to make this application as much as student-centric as possible. This would require us to think of various scenarios in which a student would use our application. We also plan on improving the UI to make the application look more polished and thorough. We also plan on writing the code to differentiate between NFC tags by reading and writing to them, as necessary.

Component Architecture Diagram



Annotations:

The component diagram essentially has a main Centralized Controller Logic that determines what action the user wishes to do. If the user taps an NFC tag, they have essentially “read” it and they will be directed to the left side of the centralized logic component. If the user wishes to add a new NFC tag they would be directed to the right side of the centralized logic component. We decided to use this workflow because it makes the most sense logically.

We intend on using the Client Server Architecture scheme for our project. In this case, the client is the mobile phone that we will be running the application on. We intend on using the Firebase database as of now to store data about user’s login and authentication information. The physical server on which our Firebase database would be located in the server in our scenario. Another “server” in our case would be the

The components that would be on the client is the NFC Front End System, TAPP processor, User Authentication, Check User Data, the Centralized Controller Logic, reading a sensed NFC tag, Reading data from DB, Performing the corresponding tasks, the local Task Database, generating an NFC ID and Writing an NFC tag.

The components which would be on the server is the Users Database which will be on Google Firebase and the NFC tag which will be placed at the desired location.

The connector between the client and server components happens through TCP/IP.