**Project Proposal**

**Introduction**

I proposed building an android application that will automate the process of signing for a class in the university. The application will have two goals: to secure the process such that there is little room for cheating or circumventing it in such a way that one person signs for two or more people and to make the process as fast and convenient as possible. In order to secure the process I propose making use of blockchain technology to keep track of changes to the data make any inconsistencies obvious. Since this is an android application that is a project for class it assumed that use cases such as those for users lacking a smartphone or unable to sign for class using the application will be dealt with later (might involve creation of a website or desktop application).

**Detailed Features and User Stories**

*Basic Functionality: Attendance check.* The app records the minimum number of classes that a student needs to attend for every course. It also sends a notification to the student of how many classes the student can afford to miss for each class 30 minutes before the class starts. If the minimum number of classes the student can miss for a course reaches 1 then the student is sent a warning every morning of the day he/she takes the particular course. If the minimum number of classes a student can miss for a course is exhausted then the sign in feature becomes inactive for the particular student and the app tells the student to go see the lecturer.

*Basic Functionality: In-Class Sign In.* This scenario assumes the lecturer is in class with the students. There are two categories of users in this scenario i.e. student and lecturer. The lecturer starts up the session for signing for class by pressing the button on the application from his/her view (preferably at the end of the class) thereby allowing the students to sign for the class in the application from there view. This is such that the students cannot sign for the class unless the lecturer has pressed the button from his/her view, they can only view their attendance record. Once the session is open, the details of each student that signs for the class including his/her photo is uploaded to the app in the lecturer’s view such that the lecturer can compare the photos/details with the actual students in class. Hence the lecturer can remove a ‘sign for class’ for any student in the session but must accompany this with a reason since this will be recorded in the blockchain ledger. In this scenario the lecturer decides when to close the session. The application also contains an alternative sign in which is the scanning of a QR code as each student user comes equipped with their unique individual QR code once they register as a student and once enrolled into the class the QR code is added to the class’ database. During the sign in the QR code is cross-referenced with those in the database.

*Basic Functionality: Remote Sign In (Whole Class).* This scenario assumes the lecturer is not in class with the students but still wants the students to sign for the class. Here signing for class will be taken care of by the lecturer. This is such that the lecturer will choose to miss class (presses button) and then give a reason for doing so and choose whether to give an assignment (presses button) or add an extra class (presses button). This facilitate the process of a lecturer, not wanting to waste a class but also not being able to make it, sends an assignment to be completed in class and handed in within the class’s allocated time. Hence the preferred time for the lecturer to open the session in this scenario is at the beginning of the class’s allocated time. Again in this scenario the lecturer decides when to close the session.

*Basic Functionality: Remote Sign In (Single Student).* This scenario assumes a student has missed a class (hasn’t signed for the class in the app before the lecturer closed the session) but has a legitimate reason for doing so and has thus requested the lecturer to edit his/her attendance record. Hence there is only category of user in this scenario i.e. the lecturer. Here the lecturer edits the particular student’s attendance record. Note that this edit will be recorded in the blockchain ledger. Generally the lecturer has the ability to edit any of the students’ attendance records but any changes to any of these records other than the initial creation of the record will be recorded in the blockchain ledger. For this reason, any changes to any of the attendance records by the lecturer must be accompanied with a reason for the change. The lecturer can, however, choose to leave the space for the reason empty.

*Basic Functionality: Holidays and Other Special Sign Ins*. All public holidays and other university special days will be coded into the app such that they won’t require signing for class. Furthermore, the user category: registrar will have the ability to set the class days for a particular class for the whole semester such that students can only sign for class on class days and not on any other days. The lecturer can, however, exclude a class day from the attendance record in the app while providing reasons. The lecturer may also add a makeup class and take attendance for it should need be if a previous class was postponed or missed for any reason at the lecturers discretion.

*Basic Functionality: Create account.* This scenario assumes that all the details that need to be entered by the registrar about each course, student and lecturer are already in the app. For a student, the app will provide an interface for entering the following details:

* student name
* student number
* phone number
* student picture
* a password

The app will also search for the student in its database of students previously entered by the registrar using the student number before completing the account creation. If the student is not found in the database then the app will stop the account creation without saving any data and ask the student to consult the registrar. The student can also edit their profile (some details and not others) after its creation. For a lecturer, the app will provide an interface for entering the following details:

* lecturer name
* lecturer phone number
* lecturer’s staff id
* lecturer’s picture
* a password

The app will also check the lecturer’s details against those already in the database and if the lecturer is not found then they are asked to see the registrar.

*Basic Functionality: Login*. This scenario assumes the user has already created an account in the app. For a student the app checks the login details before proceeding i.e. student number and password. It also checks the geo-location info of the sign in and sends this to the server. If the student never logs out then the app will remain logged in until he/she does so. For the lecturer the login simply checks the lecturer’s staff id and his/her password and the geolocation is also stored. All users are also informed of the minimum requirements for the class’ attendance and there’s a warning when the student has exceeded the limit and thus the warning sent to the lecturer and student that the student qualifies to get an E grade and that the student to see the lecturer for any necessary changes to the records e.g. dismissal from the class

# **User Roles**

## Registrar

The registrar is responsible for keying in data about the various ‘data classes’ in the application i.e. Students, Lecturers and Courses. Also make sure each student has their unique QR code with their details such as name and student number. In particular, the registrar keys in the following for each semester:

- The courses offered in each department. For each course:

* Time and day in the week its taught
* Place its taught ie class room (app will store the Geo-location info for each classroom in the campus)
* course name
* course code
* a description of the course
* course credits
* course prerequisites
* minimum attendance required

- The students taking each course

* student number
* student name

- The lecturer teaching each course

* lecturer’s name
* lecturer’s phone number
* lecturer’s staff id

The registrar is also responsible for maintaining the blockchain that forms the attendance records of each student. This maintenance includes checking each edit of the attendance record for each student and the reason for it. Hence, when an edit of the attendance record for a student is done a notification is sent to the registrar.

## Student

The student is responsible for:

* creating his/her account (provide all necessary details)
* logging in (geotagged).
* signing for class
* viewing his/her attendance
* submit assignment (only if lecturer has allowed it)
* see lecturer in case of missed attendance (remote sign in, single student scenario)

In case of issues with the attendance or if the student cannot sign in for class then the app assumes that he/she will see the lecturer physically.

## Lecturer

The lecturer is responsible for:

* creating an account (provides all necessary details)
* logging in (geotagged)
* starting a sign in session
* stopping a sign in session
* editing the course attendance file for a specific course (at any time he/she chooses) or class (In-Class sign in scenario)
* add an extra class if need be
* give assignments to be submitted through the app
* edit student attendance and course attendance for a particular student and class (remote sign in, single student scenario)
* excluding a class day from the course attendance file (holiday/special day sign in scenario)

It is assumed that the lecturer will maintain the course attendance file throughout the semester

# **Data Entities**

The following are some of the entities for which the app stores data:

* Students
* Lecturers
* Registrar
* Courses
* Classes
* Classrooms
* Class Assignments

***Suggestions on changes***

In yellow additions need further considerations