Function Specification

Project Title: NFC Powered DCU

Supervisor : Renaat Verbruggen

Document Finished On: 2020/12/04

Student Name	Student Number
Eamon Crawford	16437394
Sam O'Leary	16358763

Functional Specification Contents

0. Table of contents

1. Introduction

1.1 Overview

Our system is an amenities management system. It aims to provide users with real time info on which amenities are currently engaged, while also providing the system operator with an added level of security for these amenities and a bank of data based on user interactions.

This will allow users to quickly and easily identify the availability of their desired amenities, from a remote location, while also being able to see any upcoming bookings made for them. This information will be displayed graphically on our companion app/website.-

The NFC tag system will allow the systems operator to more easily restrict usage to authorized users, while also generating usage logs with which to analyse amenities for : popularity, rush times, and depreciation.(in the interest of optimisation).

The overall design for the system will be modular to allow for the user to adapt system functionalities to fit their needs or build for their unique implementation requirements. We will be following a microservices architecture.

1.2 Business Context

There are two business contexts we have identified for our system.

1. University service management:
The system could be implemented on the DCU Glasnevin campus for management of its computer labs, parking lots, free study rooms, and attendance. This would allow students and staff to easily identify which computer labs or parking spaces are full without having to physically check.

2. Corporation service management:
The system could be used to manage short term booking meeting rooms and parking within a company's site. This would allow staff to quickly identify free meeting rooms/workspaces.

1.3 Glossary

NFC (Near Field Communication): Is a set of communication protocols for communicating between two electronic devices over a short distance (4cm or less).

UI (User Interface): Is the contact for human users to interact with the system.

UID (User Identifier): Is a unique non-empty string that is assigned to each user to uniquely identify them within the system.

API (Application Programming Interface): Is an intermediary software that allows systems to communicate with each other.

Modular system: A system that is divided into smaller subsystems called modules which can be independently created, edited, removed, or replaced.

Microservices Architecture: variant of the service-oriented architecture (SOA) structural style – arranges an application as a collection of loosely coupled services. In a microservices architecture, services are fine-grained and the protocols are lightweight.

RFID (Radio-Frequency Identification): Is an automatic identification and data capture method which operates through using ultra-high frequency radio waves for identifying objects.

REST (Representational State Transfer): Is a software architecture style that defines a set of constraints for creating web services which allows for interoperability between like systems on the internet.

React: Is a JavaScript library for building UIs or UI components.

JSX (JavaScript XML): Is an extension for React which allows for the writing of XML type codes (like HTML) within React.

ES6 (ECMAScript 6): Is a programming language that is JavaScript standard meant to ensure the interoperability of web pages across the different web browsers.

Kotlin: Is a programming language designed for the Java Virtual Machine and Android, it is the preferred language for Android app development due to its pragmatism and lightweight design.

Postgresql: Is a more advanced version of Structured Query Language. It is an object-relational database that allows for more extensibility.

2. General Description

2.1 Product / System Functions

The general functionalities of our system are:

- Control access through NFC.
- Record UID tag.
- View availability of amenities. (status)?
- View upcoming bookings for specific amenities. (Status) ?
- Check the recorded UID list.
- View User interaction data.
- Data analytics.
- Modularly expandable.

This is a provisional list for identified core functionalities, more functionalities may be added later into development due to requirement or our own discretion.

2.2 User Characteristics and Objectives

Users of our system can be split into two separate groupings: standard users, and system operators.

Firstly, standard users:
These users will mainly utilise our system as a way of identifying what rooms/services are free as a way of directing where they will be going. Most users are accustomed to smartphones and mobile phone apps but have varying levels of computer literacy, as a result the information we display will need to be graphically concise while remaining digestible.

Secondly, system operators:
These users will be utilizing the system to simplify tasks and jobs, reducing the time required for these jobs as well as the operating costs. An example of such a task is the taking of attendance at an event, this requires each attendee to be written in and can also suffer from user error eg: handwriting. This system simplifies this task by providing ‘tap and go’ where all the data is stored instantly in a digital format.

2.3 Operational Scenarios

Standard user

Check amenities status:

The user will be able to check what amenities are currently free, in use or has an upcoming reservation by checking on the mobile phone app or connected webpage.

‘Check in’:

Users will be able to tap onto a setup NFC enabled device to have their UID recorded on the connected list.

Access amenities/services:

Users will be able to tap their connected NFC tag or device to be granted access to NFC secured amenities of services.

System operators

View database:

System operators will be able to access the connected database and view the saved system data. Allowing them to see currently registered users.

Secure amenities or services with NFC pairing:

System operators will be able to connect an NFC reader to secure an amenity or service and pair it to a set array of approved UIDs. Allowing them to easily add new amenities or services to the system.

Create UID recorder:

Create an empty list and pair it to an NFC reader to record the UID of every interaction. Simple access logging/attendance recording system to provide a portable and simple alternative to paper recording.

View UID record:

View the saved data generated by the UID recorder. Display which UIDs were recorded by the recorder for a log of access or for cross-reference with a list of users for attendance.

Data analytics:

Generate analytics based on the generated user interaction data. Gain insight on which parts of the system are in the highest demand, as well as general usage of amenities or services in the interest of depreciation of asset/expiry of assets past their end of life.

Add new NFC functionality:

Expand upon the current system functionalities to service their unique requirements. Harness the modules within the system to fulfil an unique requirement.

2.4 Constraints

GDPR:

In the case of several of our proposed use cases examples, one of the major constraints will be GDPR, these functionalities will be limited by the system operator's ability to get their users to sign on for GDPR to allow them to use their personal data.

Physical testing:

Due to the current pandemic the movement of people is largely restricted to essential travel, This impacts our ability to run a practical test as it is unlikely that we will be able to implement it on a small scale in an area of DCU, like the computing labs, and also receive a substantial enough of user base due to the diminished student presence on campus.

Accessibility:

Due to the nature of this project, it must be accessible to everyone. As such this may limit our ability to fully explore and implement our idea as additional attention must be given to ensuring that new features are not a limiting factor to any of the user base.

3. Functional Requirements

3.1 NFC Card must be able to interact with an NFC device

Description
The new ‘smart’ student card’s NFC functionality must be able to be read by NFC devices running our code

Criticality
Scanning NFC cards in a user friendly way is the backbone of our project as if the system isn't easy for the user it won't be used, spoiling our data.

Technical Issues
NFC has some competing standards, we must choose what to support.

Dependencies
none

3.2 Mobile app must be able to communicate with NFC device

Description
The mobile app will use the smartphone's NFC functionality to interact with our NFC reading device in the same way as the student cards.

Criticality
Offering an alternative or backup for users to use the system in the case of a lost student card helps ensure more accurate data and is more user friendly

Technical Issues
NFC has some competing standards, we must choose what to support. Some smartphones don't support NFC, and some restrict access to its usage for third party apps.

Dependencies
Users must be able to set up the smartphone app by entering their student number

3.3 Web site must display system information for users

Description
We must provide a web page that will show real time update information for each of our systems components, for example computer lab availability and car park space availability.

Criticality
Showing real time information to our users helps them make use of the campus resources more efficiently and saves them frustration.

Technical Issues
Ensuring the most up to date information pulling from many different systems will be a challenge to efficiently manage. Furthermore ensuring the UI is user friendly and initiative is essential. The UI must scale for smaller devices.

Dependencies
NFC Card must be able to interact with an NFC device

3.4 App must display system information for users

Description
Our mobile app must show the same real time information found on our web page, for example computer lab availability and car park space availability, but in a UI designed for smaller screens

Criticality
Showing real time information to our users helps them make use of the campus resources more efficiently and saves them frustration. Having this available as an app is more user friendly then making them go to a webpage. This is one of the main features of our project

Technical Issues
Ensuring the most up to date information pulling from many different systems will be a challenge to efficiently manage. Furthermore ensuring the UI is user friendly and initiative is essential. The app must update information in an efficient manner as data can be more limited and costly for the user on mobile devices, striking a balance between most recent data and efficient network use will be challenging.

Dependencies
NFC Card must be able to interact with an NFC device

3.5 Provide insight from data analysis

Description
As we gather large amounts of data from our systems usage, we should be able to organise and present this data to users and administrators. Our system should also to automatically pull insights from the data, such as predicting near future conditions, and use these to aid the admin and users. These will be provided like the real time data on the web page and on the app.

Criticality
This is one of the main points of our project, being able to provide data driven insights to help users plan their days and for administration to plan for the future.

Technical Issues
Getting worthwhile insights from data analysis will be a challenge as we have no experience in this area.

Dependencies
Data gathered from user interactions (3.1, 3.2)

3.6 Communicate with third parties

Description
We want our project to be modular, allowing third parties to build their own components to use our system. We also want to integrate with existing systems like the library study room booking system. Therefore our project will need to communicate with third party systems, both sending and receiving data.

Criticality
We believe that while our own uses for our system provide value to our users, we think that there will be more use cases we haven't thought of. Allowing others to easily expand on what we build is a core design feature of our project.

Technical Issues
While building an API for interaction with our nfc system isn’t very challenging, organising the data from third party components and running data analysis on it will be very tricky as we have no insight into what the data is.

Dependencies
None

3.7 Allow third parties to easily add additional features and components with a modular structure

Description
Focusing on keeping a modular, lego like structure for all our components will allow further extension of our project by others in the future, to tackle problems in use cases we haven't discovered yet. It allows our project to be integrated into other institutions where administration sees a benefit to it. Ideally, we would like others to be able to make use of our projects NFC system, event system, real time data system, and data analysis system with minimal custom code.

Criticality
NFC has been described as a solution looking for a problem, so to build a plug and play solution for any use case separates out our project from any specific use case. This is a core ideal of our project.

Technical Issues
Making every component completely independent will be a challenge, we will be following a microservices architecture pattern. Writing good documentation so other developers can easily make use of our work without reading any code will be critical

Dependencies
Communicating with third parties (3.6)

3.8 Users must be able to set up the smartphone app

Description
We need some way for users to set up the smartphone app , so it has access to the same information as found on the student's student card, which will be the student number. For administration this information will be different. We will allow users to manually enter the information, or scan the card with their smartphone.

Criticality
Without this users will not be able to use their phone as a replacement for their student card.

Technical Issues
Using the smartphone reading NFC functionality, which may be interrupted by other apps on the device, such as the leap card app.

Dependencies
None

3.9 System must provide data insights from data analysis to admins and users

Description
The system will provide insights to users and admins through the webpage and app.

Criticality
Getting insights from data analysis is one of the main points of our project, therefore showing them to the users is important.

Technical Issues
Clear UI for complex ideas may be difficult to provide.

Dependencies
Data must be gathered and saved (3.1, 3.2), and have a frontend to display this information (3.3, 3.4)

3.10 Admins must be able to log on through the webpage

Description
Administration should be able to see and manage the background logs of the project.

Criticality
Not the most important part of the project but will be useful to spot errors in the system

Technical Issues
We will need to devise some secure log on for the web page.

Dependencies
3.3

3.11 The system must be able to support GDPR opted in users, and those not opted in

Description
Certain uses for our system may require sensitive user information, for example the covid-19 contact tracing will need to identify users. Other third party components may need access to this too. Therefore we need to get permission to store this information from users, while also still supporting users who don’t opt in, so they can use the features that don’t require it.

Criticality
Our project cannot be put into production if it is not fully GDPR compliant, therefore this is critical or we lose a lot of functionality

Technical Issues
We must familiarise ourselves with GDPR as it relates to our project.

Dependencies
None

3.12 Provide covid contact tracing support, in conjunction with the current system

Description
Because our system knows when a person is on campus, and may know where they are on campus, we can give more accurate covid19 contact tracing information, if a student reports they have tested positive. We will need a report option in our smartphone app. We will only be able to use this for students who have opted in for our GDPR information dependent optional features.

Criticality
DCU already has a system for this, replacing it isn't necessary but we believe we could do better with the information we have, rather than entirely manually tracing a person's movements.

Technical Issues
GDPR concerns. Organising and analysing data to know where and when a user was at each location.

Dependencies
Gathering data and analysing it (3.1, 3.2, 3.5)

3.13 System must use its features to provide a real time system for computer lab availability

Description
We will use the data we gather from NFC devices at each computer or at each entrance of the lab, where users will scan their card to check into or out of the lab. Our system will use this information, along with DCUs timetable, to inform users of free study spaces in each lab, instead of having students interrupt rooms looking for space. We will also provide insights into which labs are busy and when, based on insights pulled from our data analysis. This data will be presented to users through the web page and mobile app.

Criticality
Computing students are more likely to adopt new technologies, so this is an ideal starting implementation of our system

Technical Issues
Multiple entrances to some labs means we need to group certain NFC devices together in our system to notice when a student leaves a different way to when they came in. Acquiring information from DCUs timetable may be challenging due to the change in system this year.

Dependencies
Data must be gathered and saved (3.1, 3.2), and have a frontend to display this information (3.3, 3.4)

3.14 System must use its features to provide a real time system for car park space availability

Description
For the car park, instead of using infrared to scan a barcode could, we can use NFC which is more reliable. The information we pull from the user's card will be used, like the current system, to check if a user has bought a car park pass or not. Of course we would continue to support the old ticket paper system as well. This would require our system to talk to DCU's system to know if someone has paid for parking or not. We would also ideally have proximity sensors on each car park space, which we can use to provide real time information through our web page and mobile app about what exact spaces are available. This information could also be displayed at the front of the car park on a monitor.

Criticality
Saving users time driving around the car parks and makes the whole system more efficient as drivers won't clog up already full car parks. It is an interesting example of implementing our system and is important for the project. The data analysis from this would be very useful for DCU to plan out timetables and judge capacity.

Technical Issues
Implementing proximity sensors or mocking them for our demonstration. Tracking the real time data accurately. MAking sure the UI is concise and fast, so users cause delay trying to find where their space is.

Dependencies
Data must be gathered and saved (3.1, 3.2), and have a frontend to display this information (3.3, 3.4)

3.15 System must use its features to provide a real time system for library study room availability

Description
The library study rooms can be booked online by students to have a space to work. These rooms are in high demand and often the current system leads to problems. This includes students booking rooms but not using them, and students taking rooms that are booked. We want to implement a check in/out system for the study rooms which will use our real time system to inform students of free rooms and when they are booked throughout the day. We will need a NFC device at each door which will sign users into the room. If someone books a room and doesn't show up we will cancel that booking. If a non booked room is taken, we will inform users it is not free. We will also run analysis on the data and inform users of the busiest and quietest times throughout the week for the study rooms.

Criticality
This would solve 2 big issues with the current system, but may require more development time due to there being no API for the current booking system, meaning we may have to replace it. It is a good use of our project but may not be worth the full time investment compared to the other implementations

Technical Issues
We will need a way of talking to DCU's current booking process, to cancel bookings on no shows, and to get the student number of the student who booked the room to match them when they enter. We dont believe this is possible with the current booking system, and therefore may have to build our own to demonstrate this. We will also need an indication on the room that it is currently booked or taken, so students don't take booked rooms. This could be as simple as a colour LED on the NFC device, or a display which shows booked times.

Dependencies
Data must be gathered and saved (3.1, 3.2), and have a frontend to display this information (3.3, 3.4)

3.16 System must use its features to provide a check in system for lectures and exams

Description
Some lecturers like to take attendance for each lecture by passing a sheet sheet of paper to sign in on. Like for each exam in DCU, exam attendants must take the student number of every student present. Both of these processes are slow and prone to human error. Instead, students could scan in with their NFC student cards, which removes the chance of human error. Ideally these devices would be portable and battery powered.

Criticality
Being able to run data analysis on student attendance could give the facility great tools to improve student learnings. Further move this goes hand in hand with our Covid19 contact tracing component.

Technical Issues
For project demonstration, acquiring suitable portable devices may be infeasible due to cost, therefore we may use a standard computer or laptop.

Dependencies
Data must be gathered and saved (3.1, 3.2), and have a frontend to display this information (3.3, 3.4)

4. System Architecture

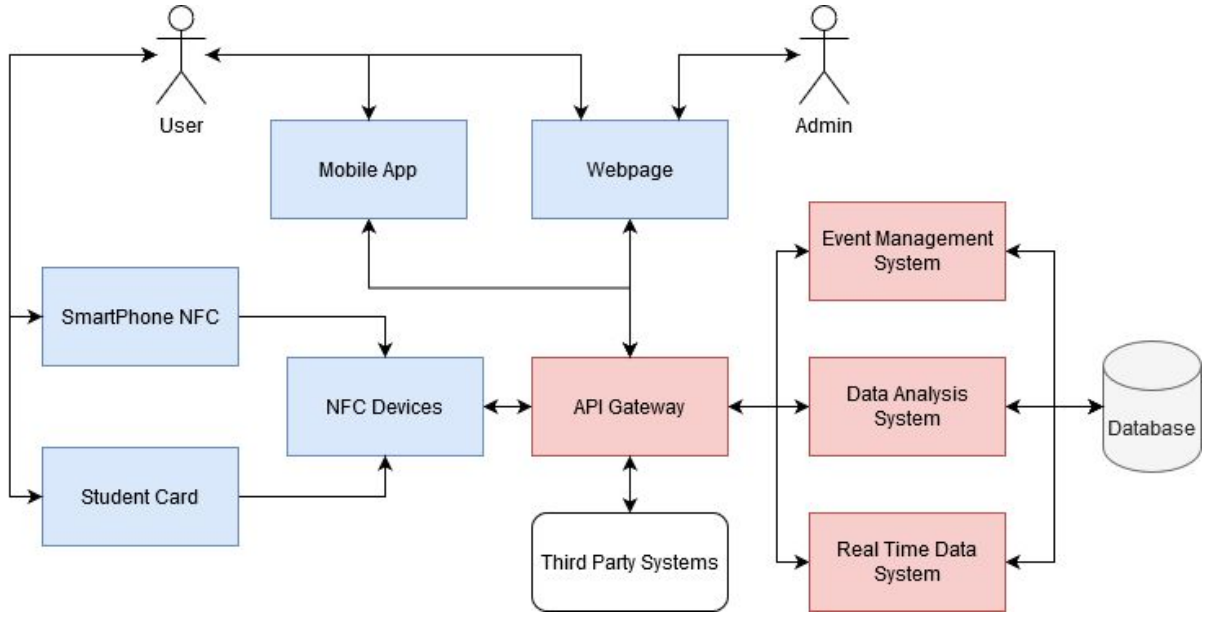


Fig. 4.1

4.2 System Architecture Diagram Description

4.2.1 Frontend

Mobile App

The mobile app will act as a user interface, providing real time data to the user, along with data analysis insights. It will acquire this information from our API Gateway service, using REST Hooks to avoid wasteful polling.

The application will be built using react native, which in theory has the benefit of writing code that will run on both Android and IOS, in practise there may be some custom code for both. React Native. The React Native framework uses JSX and ES6.

The mobile app will also act as an alternative NFC card if the smartphone has the capability to do so, this is discussed under SmartPhone NFC below.

Web Page

The web page will host a similar interface as the mobile app, again using the a REST Hook to stay updated with the Gateway API. The Webpage will also support an admin log on, where admins can view the logs of all of the systems services and manage them.

The webpage will use the React framework, which is written in JSX and ES6, with CSS for styling.

Student Card

The student card will contain an RFID NFC chip, that will store a small amount of data and will allow it to be read when in close proximity to an NFC reader. We will need to write this information to these cards once and then lock them to stop tampering.

SmartPhone NFC

If the app is installed on a smartphone With NFC capabilities, we can take advantage of this and make it act as an alternative to the Student Card.

NFC Devices

The NFC Devices will contain an NFC reader to pull information from the student cards. NFC devices may differ based on the use case it is addressing. The NFC device will call our API Gateway to interact with the rest of the system. The NFC reader will use open source drivers.

4.2.2 Backend

API Gateway

The API gateway will be the single entry point for all clients. It handles requests by routing them to the correct service or services. Any security can also be validated here. The api gateway, like all of the backend, will be written in Kotlin.

Event management system

The event management system will handle any functionality that needs to happen with a request, for example admin system commands or canceling a booking in an external service. It will be a microservice.

Real Time Data System

This system will handle organising and formatting the real time data to be present on the front end systems. It will be written in Kotlin.

Data Analysis System

This component will handle running data analysis on the historical data in the database and providing insights from this to our users. It will be written in Kotlin.

Database

Where All data will be kept. It will be queried by Postgresql.

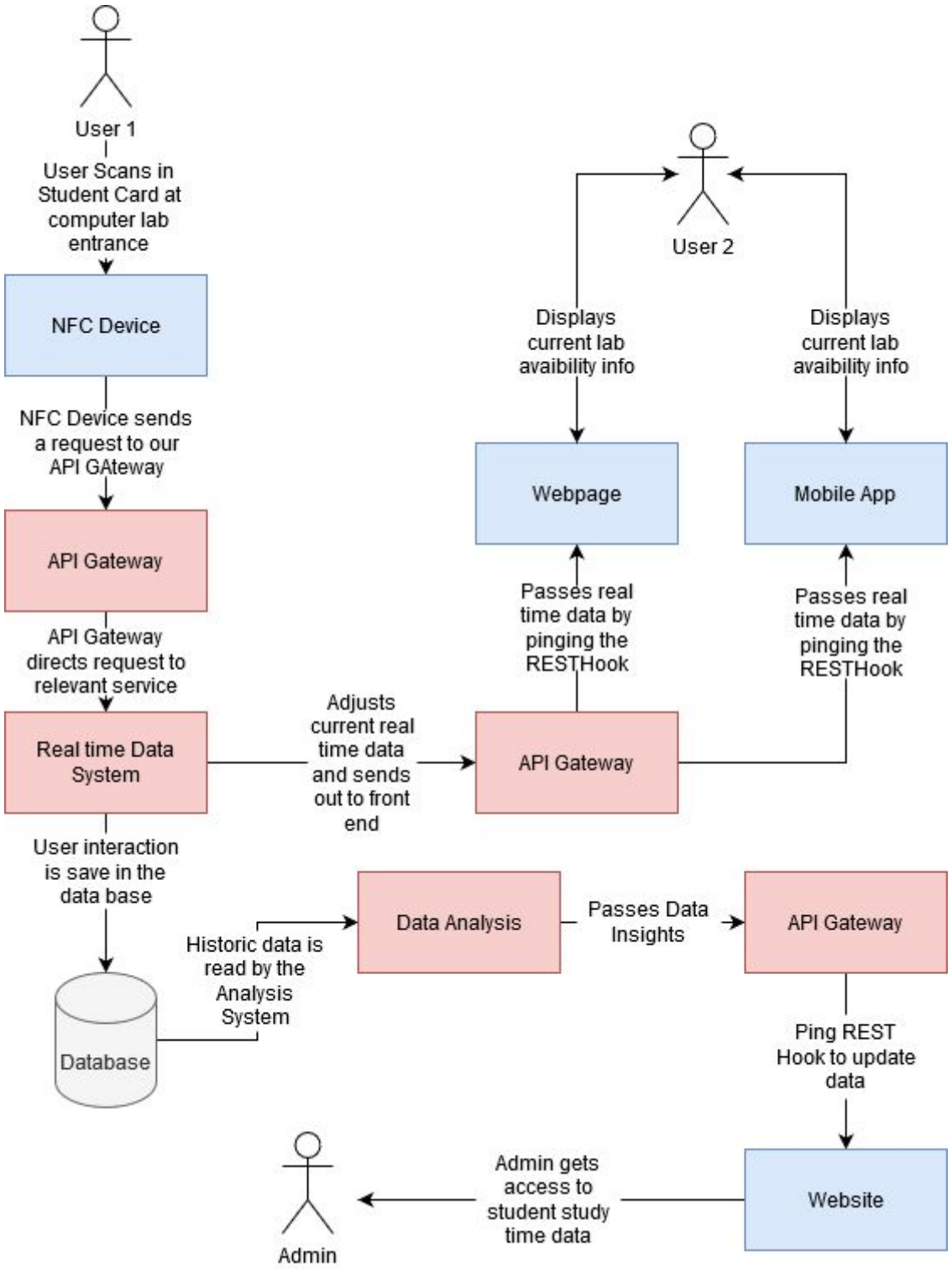
4.2.3 Other

Third party systems

Will be any systems we need to interact with, that we haven't built. Such as the dcu timetable system, the car park pass database, or any future components that wish to make use of our project.

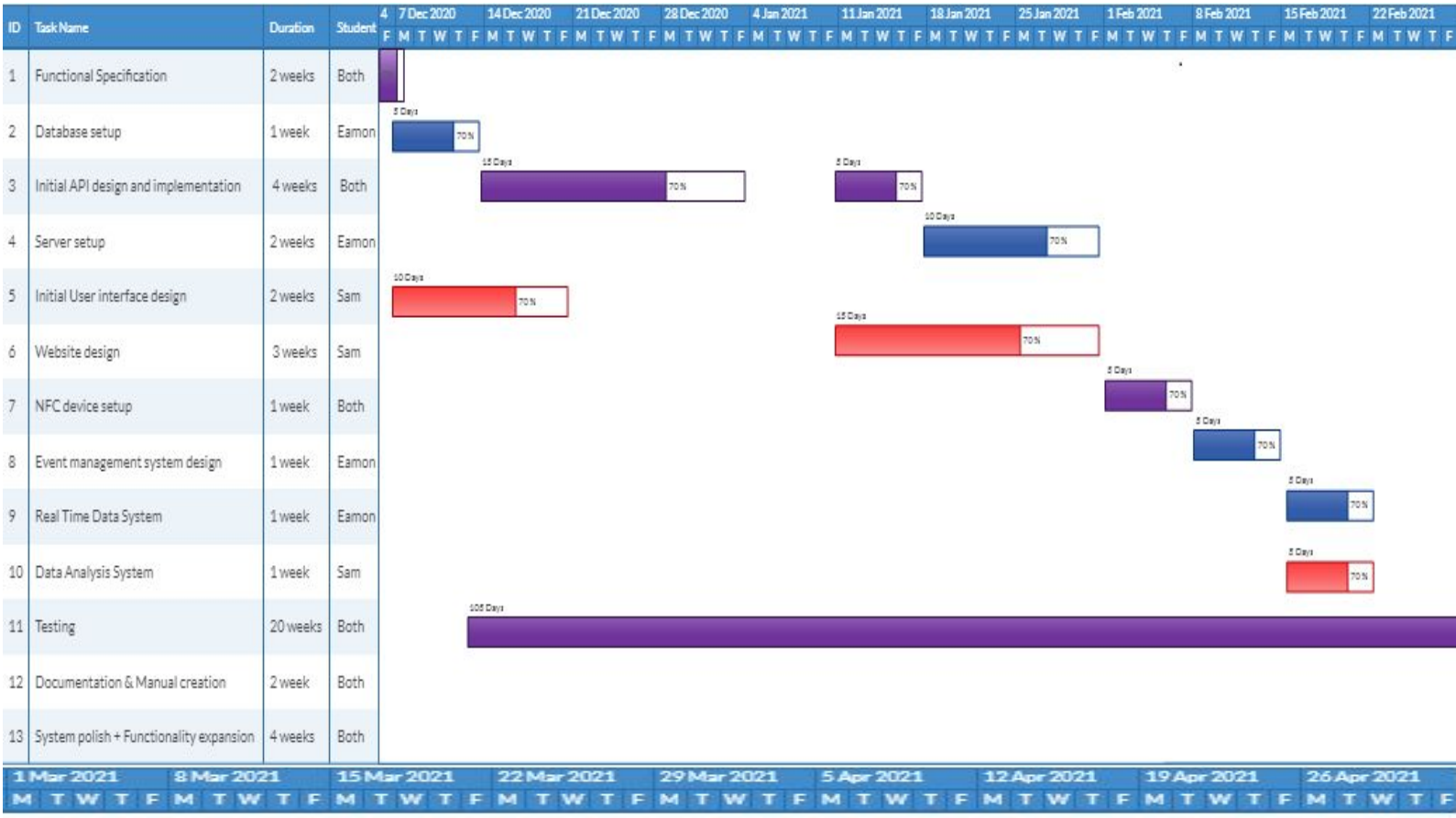
5. High-Level Design

Logical data flow of a User checking into a lab



6. Preliminary Schedule

Provided below is a preliminary schedule for the completion of this project. Included in the schedule is a 1 week break in January to focus on our semester 1 examinations, as well as 4 spare weeks of time in semester 2 to allow for errors or delays in completion due to unforeseen difficulties. These spare weeks are also to allow us to focus on assignments or exams for modules in semester 2 (as we do not yet have the dates for these we cannot specify their place within our chart.)



ID	Task Name	Duration	Student
1	Functional Specification	2 weeks	Both
2	Database setup	1 week	Eamon
3	Initial API design and implementation	4 weeks	Both
4	Server setup	2 weeks	Eamon
5	Initial User interface design	2 weeks	Sam
6	Website design	3 weeks	Sam
7	NFC device setup	1 week	Both
8	Event management system design	1 week	Eamon
9	Real Time Data System	1 week	Eamon
10	Data Analysis System	1 week	Sam
11	Testing	20 weeks	Both
12	Documentation & Manual creation	2 week	Both
13	System polish + Functionality expansion	4 weeks	Both

7. Appendices

Specifies other useful information for understanding the requirements.

Kotlin: https://kotlinlang.org/

React: https://reactjs.org/

React Native: https://reactnative.dev/

Postgresql: https://www.postgresql.org/