**Success Criteria**

* Scan the equipment to locate it in the inventory.
  + Barcodes on devices, how can this be implemented for future equipment?

Within our system, we have created a database storing the inventory and hosted this on the server 10.74.1.60 to give ourselves some type of medium to interact with. Originally we mocked a database using Microsoft Access to give us some insight into how we may potentially go about devising the rest of the system flow process; from there, we uploaded a Microsoft Access file using ODBC transfer to an SQL Server on the 10.74.1.60 server. We have implemented Firebase MLK Libraries in our mobile app which gives barcode scanning functionality. We have an item ID structure which looks like follows:

TH13

TH = Thornton Campus 🡪 The idea is that this app in the future could be transferable to other campus’ e.g. Parkgate campus may use the PG prefix.

1 = Device Type Reassurance\* 🡪 The first number in the code refers to what the device type is. In this case, 1 stands for Laptops whereas 2 would stand for Microphones.

3 = Device Number 🡪 This is the number of the device type. So, this item would be the 3rd laptop in accessibility.

So in this case, TH13 stands for Thornton Device which is a Laptop which is 3rd in storage.

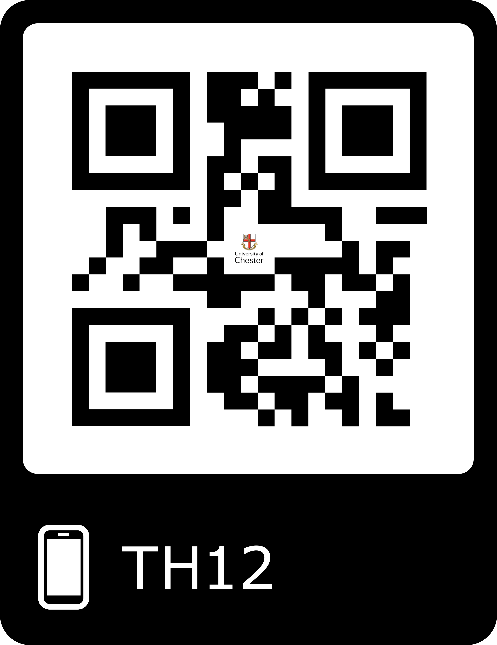
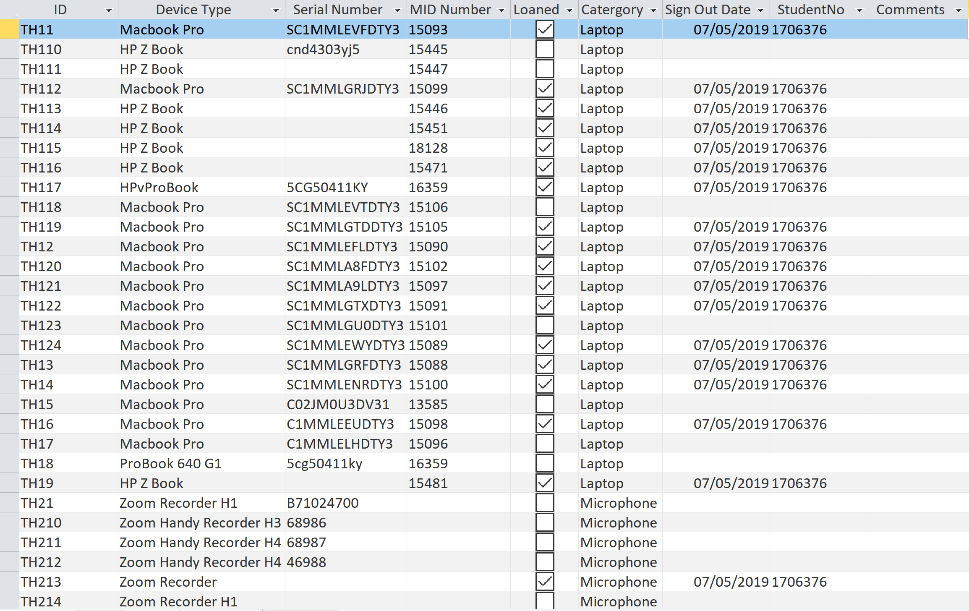
In another case, TH32 stands for Thornton Device which is a Camera which is 2nd in storage.

Using this structure, we believe once understood that this can be implemented easily in the future, creating barcodes following this structure. Without the need for technical support.

\*Device Type Reassurance number codes are as follows:

|  |  |
| --- | --- |
| Number Value | Device Type |
| 1 | Laptops |
| 2 | Microphones |
| 3 | Cameras |
| 4 | VR |
| 5 | Tablet |
| 6 | Console |
| 7 | PC |

We decided as a group that we were going to use QR codes instead of barcodes in this instance. We felt from a user experience standpoint that QR codes would be easier as you do not necessarily have to scan it from one specific angle because it has fixed corner positioning so it is clever enough to adjust itself (Pardede, 2010), used as a quick fix added feature as well as reasoning behind why we opted to go down this route. For potential failure of scanned QR Codes, we have also included the ID number underneath just in case and these will also be used on the backend web app system.



* After selecting the item display a pop up message containing terms and conditions (click ok to continue).

In the app, once the user has signed in which also requires them to scan their student card barcode in order to log in (Trimming ‘B’ at the beginning and ‘101’ from the end of the barcode to get their student number e.g. typical barcode = B1706376101), and then once the specific QR code has been scanned for the relevant device, this will display a pop up box displaying all the terms and conditions as well as the option to agree or disagree to the terms. If the user has agreed to the terms, then the item will be loaned out to them however if they choose to disagree then they will be brought back to their original screen. Once agreed, the user will have loaned out that item.

* Scan student cards and automatically set a two week loan (Paul can override this to a custom length of time)

Students are required to scan their student cards as mentioned previously using their barcode. We have a system in place that once the user has accepted the terms, then the device will automatically be registered out in their name. We register the Signed Out date and have a Mail Service in place reminding them about the expiry of their loan.

* Note/comments section with a free text area and tick boxes for includes adapter, USB cable and laptop case

We have a comments system in place only on the back-end web/desktop application. This can only be accessed by this user to attach notes and comments to an item.

* Send a confirmation email with the return date

N/A. We have spoken with Paul and his response was that it is not exactly necessary because emails are sent 2 days prior to the return date so our progress up to this point covers this functionality.

* When card is scanned or student number is entered display picture of user e.g. using the link below and changing the student id

E.g. using the link below

[https://apps.chester.ac.uk/v2/profile/profile.php?operation=photo&otheruser=**1024861**](https://apps.chester.ac.uk/v2/profile/profile.php?operation=photo&otheruser=1024861)

When the user scans their student card, the Student Number string was appended to the above link to generate an image of the user on the student card for extra security should Paul need to check the identity of the student.

* Automatically send emails asking them to return the equipment when close to the deadline(or renew the loan, this can only be done once)

We have created a Mail Service that runs every 2 days that will check to see if each record in the database has a sign out date field not null. If this sign out date field is not null, then a Recurring Job using HangFire will be triggered, sending an email to the user to notify them that they must return their item. This code is constantly running with the API.

* Send a reminder email every two days if overdue

As mentioned previously, the Mail Service will run every two days informing students if their loan is overdue until the sign out date field returns null which occurs once they sign the item back in.

* The system would need to allow Paul to manually change a device status e.g. if it is out of order and not loanable

The web/desktop app has a feature that allows Paul to manually change data displayed in the database. Should extensions to loans be required then Paul can edit this within the web app.

**Desirable Feature**

We created an API that links to the database. With this, as a long term goal, now that the API has been created, it may be used in the FSE.NETWORK to be used however it is wished.

**References:**

Pardede, E. (2010). *Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia* (p. 430). New York, NY: ACM.