Jonathan Barry

MOBILE APPLICATION report

B8IT120

* Topic and objectives of your Mobile Application

As per the project brief, I set out to design a Dublin bikes app that renders the API data from JcDecaux on a map that will update the info of each station with the number of bikes and free spots in real time. It will provide a registration and log-in feature, will then track the journeys that a user takes, allow them to attach pictures for each journey, and also allow them to view their historical journeys. I want this to be as seamless, functional, and quick as possible for the user.

* Target Audience

I’m attempting to target any users of Dublin bikes. This demographic covers a pretty vast swathe of the population – both myself and my sixty-seven year old father frequently use Dublin bikes for instance, and you only have to look at anyone cycling by on one to see that a varied segment of the population takes advantage of this service. Indeed, with nearly seventy thousand annual subscribers, two million journeys this year alone, and eighteen thousand journeys recorded on a single day last year, the bikes have very broad appeal and thus, any app that targets its userbase should also have broad appeal, and my app should be accessible to all (Dublin Bikes, 2017).

* Rationale behind Development Approach (hybrid or native)

It was frequently pointed out during class that Zuckerberg had said that they’d “wasted two years” (The Verge 2017) developing hybrid apps, and that developing natively was now the way of the future.

Delving further, I read that “[t]he biggest hangup surrounding hybrid apps has traditionally been that the JavaScript connections required for an app to access device functions — like the camera, GPS, etc. — could be resource intensive and slow down an app, giving it a stuttery or laggy feel”. With my app using the google maps API along with the device’s camera, I wanted to this app to feel responsive, quick and smooth when using the phone’s wider functionality which is integral to my app. By developing it natively, I would enable those who don’t have powerful phones to use my app comfortably as it is often said that Hybrid apps are fine to develop as long as the end-users’ phones are powerful.

This is also a relatively simple app, so while it arguably took a higher investment of time and resources (a criticism levelled at developing native apps in comparison to hybrid), ultimately the difference between the two development approaches was negligible. Furthermore, moving forward, this app should be easier to maintain and upgrade.

* Description of the cloud services used to manage back end (registration, login, profiles, image storage)

I used Google’s firebase for user authentication and storage of images and journeys. Each user has to register before using the app, logging their details in firebase, and I then used these details to authenticate them upon logging in. By default, I have the app greet them by displaying their e-mail in a welcome note, but when they update their name and address on a separate profile page which is then stored in the database, the welcome note then changes to include their name instead.

Furthermore, each journey they save is attached to their unique user ID (as determined by firebase) so they can only see the journeys that they have taken and nobody else’s.

Each image that they upload is stored in Firebase storage, and upon uploading the image, I retain its unique URL that is then saved as a string in the firebase database as part of the journey object, and it is then called each time they load up their historical journeys. However, uploading a photo is not mandatory when logging a journey.

* Differences between prototype and final deliverable and an explanation of the differences

My prototype was quite basic, and everything was pretty separate from each other. For example, the JSON data was only rendering to a listadapter, and the image storage wasn’t linked to the database. However, by persisting, I was able to have everything working in harmony.

* A link to your original prototype

Unfortunately this isn’t currently available, but I can assure you that it was rather poor.

* Technologies/Techniques used (may be covered in development approach and cloud services used)

To fetch the real-time data of Dublin bikes, I parsed the JSON data that’s freely available from the JCDecaux API. Then, I rendered it on the page by looping through the collected JSON objects and populating markers with their unique details including its name, spaces available and bikes available.

These markers are accessed when a user goes to add a journey and the map screen is brought up – I set up onclick listeners on each marker so that when a user clicks on a marker, an info window is brought up with number of spaces available, number of bikes available, and the station’s name. Then when the user clicks on that info window, it is selected as the starting point of that journey, and then the user repeats the process for an end point (this is controlled by a simple if-else statement that has a Boolean). The user can tell whether they’re choosing their start or end point as I have a floating text view hovering over the map which tells them. Toast messages also pop up to let the user know when they have selected a station.

Then, these two station objects are passed back into the add journey screen and the user can see at what point they started and ended at. Here, they can also add a picture to their journey and when selecting this option, the phone’s camera is started and the user can take a picture. As I have explained earlier, this is then uploaded to firebase storage, and its unique URL is stored as a string in a journey object. If the user is now happy with everything, they can then add the journey and the date is automatically recorded for them when uploading the journey object to the firebase database. They can do this as many times as they like to add as many journeys as they like.

Then, to view historical journeys, the app fetches the journeys that are associated with that particular firebase user ID. These journeys are then rendered on a page by using a listadapter of journey objects that displays the start point, the end point, the date of the journey, and the picture associated with that journey (if present). The user can then click on any one of these journeys in the list, and the app will then send the user to google maps with only the markers of that journey present. These markers are different colours to differentiate between the start and end points (and it is also written on the markers as well).

There’s an options menu on each page to increase accessibility and allow the user to jump to any page that they wish. Also, I have each string saved in the strings.xml file for ease of translation. I used Picasso to render images in my app.

* Strengths and weaknesses of your Mobile App Application

Well, it achieves what was defined in the specification but there are certainly improvements to be made. I’m pleased at the app’s speed when rendering data and switching between pages. I also think it’s intuitive and easy to understand for all users.

As for weaknesses, I think that the UI leaves a lot to be desired, it’s quite blocky and plain, and I did not do a lot of customization (apart from some cursory tinkering with the custom map markers). So nicer fonts, nicer buttons, and a more generally aesthetically pleasing layout would be a marked improvement. This is especially evident in the layout of the historical journeys page which is horrendously ugly in my opinion. There are is also no way to view the picture as a larger image when scrolling through the list of journeys, instead it just goes to the map markers directly.

Functionality wise, a route tracker which would join the two markers would be a great improvement. Also, allowing the user to modify their start and end points a little more would be beneficial. Currently, the app operates by forcing the user to select a marker to begin with, then they must select marker to finish with, and then the selection process is over and they’re brought back to the add journey screen. The user is not locked into this selection when adding a journey, and they can re-do this process easily, but I would like the user to have been able to alter their start point while within the maps activity.

Also, when uploading images, the user (if they are quick) can successfully take a picture and add the journey to the database without the image properly uploading. I found it difficult to prevent this, as I really do not want to make uploading pictures mandatory. Finally, the option to delete a journey would be a good addition.

* References/Bibliography

Aplexa AB (2017) ‘Hybrid vs. Native in 2017 — (And Why We Based Our Business On Cross-Platform)’ Available at

<https://medium.com/appademics-magazine/hybrid-vs-native-in-2017-and-why-we-based-our-business-on-cross-platform-100d12b1030a>

(Accessed: 25 September 2017).

Dublin Bikes (2017) ‘Just eat DublinBikes – latest figures!’ Available at

<http://www.dublinbikes.ie/Magazine/Reports/Just-Eat-dublinbikes-latest-figures>

(Accessed: 25 September 2017).

The Verge (2012) ‘Mark Zuckerberg promises a native Android app, says betting on HTML5 for mobile was a 'mistake’’. Available at: <https://www.theverge.com/2012/9/11/3317230/mark-zuckerberg-betting-on-html5-for-mobile-was-a-mistake-hints-at>

(Accessed: 25 September 2017).

* Appendix

26/09/17 - Firebase authentication, database, camera activity and uploading to firebase

27/09/17 – Get maps activity, JSON data successfully rendering as text, begin to save things in firebase

28/09/17 – Getting markers on maps (JSON data)

29/09/17 – Saving markers

30/09/17 - Historical journeys with maps, image association

01/10/2017 – Strings, UI design.