Catanduanes State University
College of Information and Communications Technology
Virac, Catanduanes

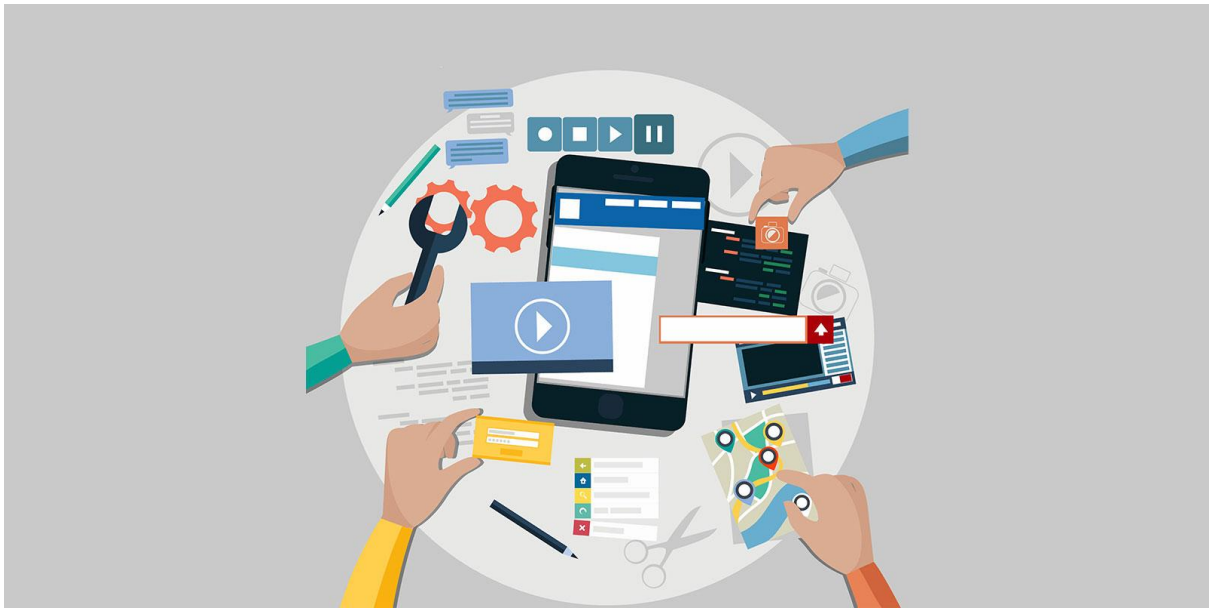


Image from: https://buildfire.com/hybrid-vs-native-mobile-app-development-better-data-driven-answer/

LEARNING MATERIALS AND COMPILATION OF LECTURES/ACTIVITIES

ITRACKA3

Platform Technologies: Data-Driven Mobile Application

DISCLAIMER

This learning material is used in compliance with the flexible teaching-learning approach espoused by CHED in response to the pandemic that has globally affected educational institutions. Authors and publishers of the contents are well acknowledged. As such, the college and its faculty do not claim ownership of all sourced information. This learning material will solely be used for instructional purposes not for commercialization.

CatSU College of Information and Communications Technology

TABLE OF CONTENTS

CHAPTER 1: OVERVIEW OF MOBILE APPLICATION DEVELOPMENT

LEARNING OUTCOMES

At the end of this chapter, the students shall be able to:

1. Understand the basic concepts of mobile application development.
2. Recall terms and app ideas from previous course.
3. Differentiate Native vs Hybrid vs Web applications.

KEY TERMS

1. Data Storage
2. Data-Driven Apps
3. Native Apps
4. Hybrid Apps
5. Web Apps
6. Front-End Development
7. Back-End Development
8. Preferences
9. Databases
10. App-Specific Storage

LESSON 1: INTRODUCTION TO MOBILE APPLICATIONS

Today mobile device users prefer to use applications installed on their smartphones. They use installed applications (apps) for carrying out routine activities like booking cabs, buying movie tickets, and watching videos on YouTube. This trend is confirmed by Gartner, which has this to say: “Enterprises are finding that they need to support multiple platforms, especially as the BYOD [bring your own device] trend gains momentum.”

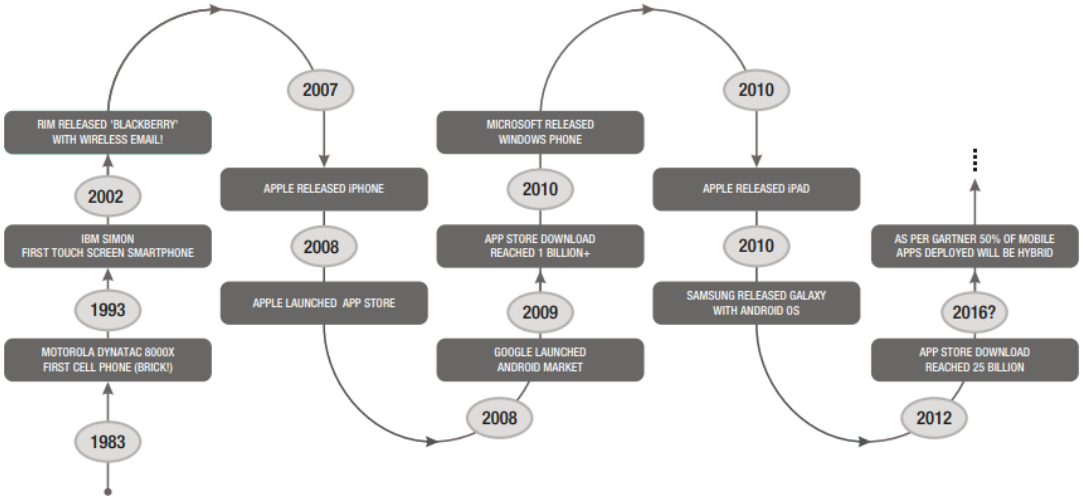
In addition, the app-only trend, set by e-commerce companies like Flipkart and Myntra indirectly supports this BYOD trend. App-only means that an application, which used to be available through a desktop web browser as well as a mobile device, stops operation of the web application, forcing the customer to access the application through a mobile app only.

The market share of mobile devices is divided mainly into Android, iOS, and Windows Phone. Because of the differences in platforms/operating systems, creating an installable mobile application that targets multiple device platforms requires too much of effort and expertise. For example, you have to write code in Java for Android, in Objective C for iOS, and in .NET for the Windows Phone. Shortcomings of this development approach are as follows:

- More development time
- Different expertise required per platform
- Considerably high cost of development

Now, since you have developed Android applications in your prerequisite course, we will continue developing them in this course, focusing on proper presentations of data (data-driven applications).

1.1. THE EVOLUTION OF MOBILE APPLICATION DEVELOPMENT



In 1983 came the Motorola DynaTAC 8000X cell phone—the first commercial cell phone. People called it a brick because of its 2.5 pound weight! It was sold at the price of $3,995! This phone could do little more than calling.

The first innovations came from Nokia and other manufacturers who took this technology to another level by adding more functionality, including games such as Snake and Ping Pong. In the 1980s and ’90s, mobile users had few offerings to choose from device vendors. Mobile applications were limited to those preinstalled by vendors. However, some vendors did offer applications using the Wireless Application Protocol (WAP). These applications were available from the phone manufacturers.

In November 1993, IBM launched Simon. It had preinstalled applications such as a calendar, a clock, a calculator, a notepad, and email. It also had a stylus!

1.2. MOBILE APPLICATIONS AND DEVICE PLATFORMS

There are two dominant platforms in the modern smartphone market. One is the iOS platform from Apple Inc. The iOS platform is the operating system that powers Apple's popular line of iPhone smartphones. The second is Android from Google. The Android operating system is used not only by Google devices but also by many other OEMs to build their own smartphones and other smart devices.

Although there are some similarities between these two platforms when building applications, developing for iOS vs. developing for Android involves using different software development kits (SDKs) and different development toolchain. While Apple uses iOS exclusively for its own devices, Google makes Android available to other companies provided they meet specific requirements such as including certain Google applications on the devices they ship. Developers can build apps for hundreds of millions of devices by targeting both of these platforms.

1.3. MOBILE APPLICATION DEVELOPMENT LIFE CYCLE

There are two interlinked core components of a mobile application: 1) the mobile application “Front-End” that resides on the mobile device, and 2) the services “Back-End” that supports the mobile front-end.

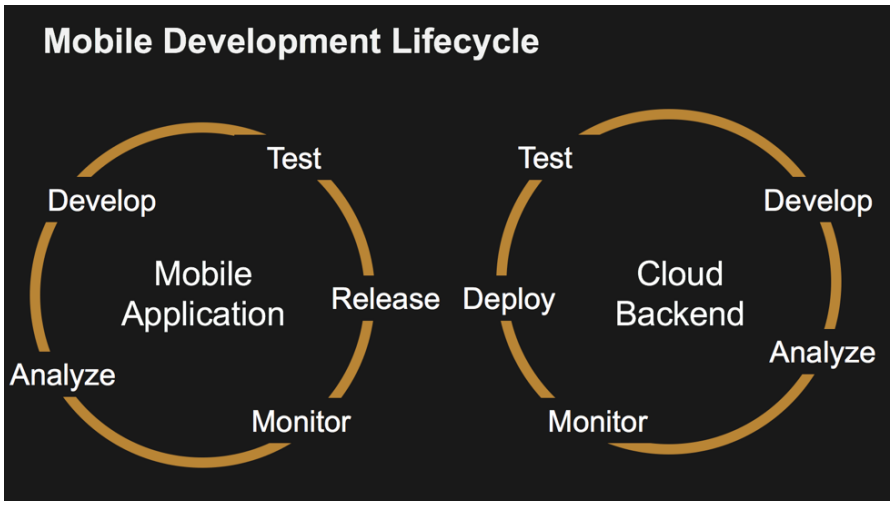


Image from: https://aws.amazon.com/mobile/mobile-application-development/

1.3.1. FRONT-END VS BACK-END

In the early days of the modern smartphone applications era, mobile applications went through a similar evolution as first websites. At first, the applications and sites where wholly contained within themselves and acted as little more than static advertisements for the brand, company, product, or service.

However, as connectivity and network capabilities improved, the applications became increasingly connected to sources of data and information that lived outside of the app itself, and the apps became increasingly dynamic as they were able to update their UI and content with data received over the network from queries to data sources.

As a result, the mobile front-end applications increasingly rely on and integrated with back-end services which provide data to be consumed through the mobile front-end. Such data can include, for example, product information for e-commerce apps or flight info for travel and reservation apps. For a mobile game, the data may include new levels or challenges and scores or avatars from other players.

1.3.2. HOW FRONT-ENDS "TALK" TO CLOUD BACK-ENDS

The mobile front-end obtains the data from the back-end via a variety of service calls such as APIs. In some cases, these APIs may be owned and operated by the same entity developing the mobile application. In other cases, the API may be controlled by a third party and access is granted to the mobile application via a commercial arrangement.

For example, a developer may obtain social media or advertising content by making calls to media or advertising company services. In this case, a developer may have to sign a contract in order to obtain credentials and a key that grants access to the API and governs how that developer can use it, how much it will cost, or how frequently it may be called, or how much data can be requested over what time period.

1.4. FRONT-END DEVELOPMENT

When considering the front end in web technologies, one term promptly comes to mind: UI design. However, the front end is much more than only the UI. You have two separate and confusing terms to understand:

- User experience (UX)
- User interface (UI)

UX is not UI. Let's take a simple example from web technologies. Your client wants to have a logo—an image—at the top of the home page. Where will you put it? If you are from Asia, based on your experience, you may place it at the top left. But it depends. Who can best solve this problem is a UX-er, also known as the client's advocate.

UX experts are always confused with UI designers. UX-ers are not UI designers! For example, while defining native ecosystem-specific mobile applications, UX-ers are better at identifying the following:

- What customers really want in the interface?
- How to provide a better experience on the small screen?
- Based on the ecosystem, how the application can achieve certification?
- Where to put which component?
- Aesthetic structure and color scheme?

A UI designer might design the interface based on the UX-er's inputs. When considering ecosystem-specific mobile applications, such as Apple and Android, the design principles differ, as shown below:

Apple Guidelines	Google / Android Guidelines
Aesthetic Integrity This represents how well an application appears. For example, in healthcare applications, the response time matters and more than the graphics. **Consistency** This confirms whether a prior user of the Apple ecosystem can use your application without any extra learning curve. Further it can confirm the following: • Is the application consistent in looking like other applications? Does it use system-provided icons and controls correctly? • Do the same icons do the thing that they are supposed to? • Is it consistent with earlier versions, if any?	**Accessibility** This ensures that the application can be used by a user with disabilities. This principle recommends including alternative approaches when a user needs to run the application under the following conditions: • Without sound • Without color • With high-contrast mode • With the screen magnified • With a combination The following shows an example of providing accessibility.

The following image shows consistency in terms of visual components.

1.5. BACK-END DEVELOPMENT

Back-end development is again a complex term. On the Web, the back end refers to the code that may communicate with a server. In a mobile application development scenario, this term can have multiple meanings. Code that you write in a mobile application may do any of the following:

- Bind data with the UI(manipulate UI components)
- Get data from the UI
- Send and receive data to and from server
- Communicate with APIs offered by the ecosystems to access sensors
- Execute only business logic without a UI (for example, a service without a UI)

This functionality can be achieved by a mix of middleware components as well as including mobile app servers via Mobile Backend as a Service (MBaaS).

SUPPLEMENTARY LEARNING RESOURCES

Mobile Application Development. (2019). Amazon Web Services, Inc. http://aws.amazon.com/mobile/mobile-application-development/

Panhale, M. (2016). *Beginning Hybrid Mobile Application Development*. Apress.

SELF-ASSESSMENT QUESTION 1

Differentiate the job of the front-end developer from the back-end developer.

LESSON 2: TYPES OF MOBILE APPLICATIONS

We see hundreds of new apps in the market every year, and the demand for mobile application development continues to increase. Although we're familiar with operating systems (i.e. iOS and

Android), chances are, we're not clued-in on the specific technology platforms software developers use throughout the design and development process in building apps.

Image from: https://www.charterglobal.com/understanding-the-3-types-of-mobile-apps-development-services/

2.1. NATIVE MOBILE APPLICATIONS

Image from: https://clevertap.com/blog/types-of-mobile-apps/

In native mobile application development (NMAD), mobile applications are developed using languages supported by the mobile OS technology stack. Before Nokia started the mobile application trend, cell phones could do only two things: make calls and send text. When a mobile application is created using the mobile OS technology stack exclusively, it is called a native application. These applications are built by using only the tools and technologies (including programming languages) suggested by the mobile application stack vendors, such as Google (Android), Apple (iOS), and Microsoft (Windows Phone).

2.1.1. PROS AND CONS OF NMAD

Native mobile application development on any platform has pros and cons. The advantages are as follows:

- Better performance
- Easier development
- Easy money-making through built-in app store sales

Here are the disadvantages:

- Increased development time and costs
- Content restrictions and guidelines, based on the ecosystem

2.2. HYBRID MOBILE APPLICATIONS

Image from: https://clevertap.com/blog/types-of-mobile-apps/

In hybrid mobile application development (HMAD), mobile applications are developed using a technology stack and are packaged to deploy on many mobile devices with different screen sizes and manufacturers. Hybrid applications allow an application developer to build an application by using simple technologies such as HTML, CSS, and JavaScript. Sometimes developers use C# and VB.NET.

Hybrid mobile applications try to mix the best of both approaches; they use the power of server-side computing but don't treat the device only as a front end. These applications have a native component that resides on the device and can use the local features as if it's a native application. That is why hybrid applications are becoming more popular than other approaches.

2.2.1. WHY HMAD?

Being able to develop once and deeply is often a motivation for using HMAD. Although we've discussed using the same code base, which seems easy, you still might need to change about 20 percent of the code, based on the platform. Why? Let's assume you are targeting Android and iOS at the same time. Sometimes the APIs used for the accelerometer, for instance, differ with respect to the platform. Some devices may not have the capability or sensor itself. Acceptance guidelines from Apple, Google, and Microsoft stores are different. Finally, UI consistency can still allow for differences on different platforms. Yet even this 20 percent of the code effort is much better than creating 100 percent of the code again. Because of this code reusability, HMAD is always better.

2.4. WEB APPLICATIONS

Image from: https://clevertap.com/blog/types-of-mobile-apps/

Web apps behave similarly to native apps but are accessed via a web browser on your mobile device. They're not standalone apps in the sense of having to download and install code into your device. They're actually responsive websites that adapt its user interface to the device the user is on. In fact, when you come across the option to "install" a web app, it often simply bookmarks the website URL on your device.

2.4.1. TECHNOLOGY USED IN WEB APPLICATIONS

Web apps are designed using HTML5, CSS, JavaScript, Ruby, and similar programming languages used for web work.

2.4.2. PROS AND CONS OF WEB APPLICATIONS

Pros: Because it's web-based, there is no need to customize to a platform or OS. This cuts down on development costs. Plus, there's nothing to download. They won't take up space on your device memory like a native app, making maintenance easier – just push the update live over the web. Users don't need to download the update at the app store.

Cons: But this is also pertinent: web apps are entirely dependent on the browser used on the device. There will be functionalities available within one browser and not available on another, possibly giving users varying experiences.

And because they're shells for websites, they won't completely work offline. Even if they have an offline mode, the device will still need an internet connection to back up the data on your device, offer up any new data, or refresh what's on screen.

SUPPLEMENTARY LEARNING RESOURCES

Panhale, M. (2016). *Beginning Hybrid Mobile Application Development*. Apress.

Valdellon, L. (2020). *What are the different types of mobile applications? And how do you choose.* Retrieved from: https://clevertap.com/blog/types-of-mobile-apps/

SELF-ASSESSMENT QUESTION 2

List your own pros and cons of the 3 types of mobile applications.

LESSON 3: INTRODUCTION TO ANDROID DATA STORAGE

Android uses a file system that's similar to disk-based file systems on other platforms. The system provides several options for you to save your app data:

1. **App-specific storage:** Store files that are meant for your app's use only, either in dedicated directories within an internal storage volume or different dedicated directories within external storage. Use the directories within internal storage to save sensitive information that other apps shouldn't access.

2. **Shared storage:** Store files that your app intends to share with other apps, including media, documents, and other files.
3. **Preferences:** Store private, primitive data in key-value pairs.
4. **Databases:** Store structured data in a private database using the Room persistence library.

3.1. CHARACTERISTICS OF ANDROID DATA STORAGE

The characteristics of these options are summarized in the following table:

	Type of Content	Access Method	Permissions needed	Can other apps access?	Files removed on app uninstallation?
App-specific files	Files meant for your app's use only	From internal storage, `getFilesDir()` or `getCacheDir()` From external storage, `getExternalFilesDir()` or `getExternalCacheDir()`	Never needed for internal storage Not needed for external storage when your app is used on devices that run Android 4.4 (API level 19) or higher	No	Yes
Media	Shareable media files (images, audio files, videos)	`MediaStore` API	`READ_EXTERNAL_STORAGE` when accessing other apps' files on Android 11 (API level 30) or higher `READ_EXTERNAL_STORAGE` or `WRITE_EXTERNAL_STORAGE` when accessing other apps' files on Android 10 (API level 29) Permissions are required for all files on Android 9 (API level 28) or lower	Yes, though the other app needs the `READ_EXTERNAL_STORAGE` permission	No
Documents and other files	Other types of shareable content, including downloaded files	Storage Access Framework	None	Yes, through the system file picker	No
App preferences	Other types of shareable content, including downloaded files	Jetpack Preferences library	None	No	Yes

Database	Structured data	Room persistence library	None	No	Yes

3.2. DATA STORAGE NEEDS

The solution you choose depends on your specific needs:

3.2.1. HOW MUCH SPACE DOES YOUR DATA REQUIRE?

Internal storage has limited space for app-specific data. Use other types of storage if you need to save a substantial amount of data.

3.2.2. HOW RELIABLE DOES DATA ACCESS NEED TO BE?

If your app's basic functionality requires certain data, such as when your app is starting up, place the data within internal storage directory or a database. App-specific files that are stored in external storage aren't always accessible because some devices allow users to remove a physical device that corresponds to external storage.

3.2.3. WHAT KIND OF DATA DO YOU NEED TO STORE?

If you have data that's only meaningful for your app, use app-specific storage. For shareable media content, use shared storage so that other apps can access the content. For structured data, use either preferences (for key-value data) or a database (for data that contains more than 2 columns).

3.2.4. SHOULD THE DATA BE PRIVATE TO YOUR APP?

When storing sensitive data—data that shouldn't be accessible from any other app—use internal storage, preferences, or a database. Internal storage has the added benefit of the data being hidden from users.

SUPPLEMENTARY LEARNING RESOURCES

Data and file storage overview. (2019). Android Developers. https://developer.android.com/training/data-storage

SELF-ASSESSMENT QUESTION 3

Why do you need to carefully choose the type of storage you are going to use? Explain your answer.

LESSON 4: BUILDING WEB APPS IN WEBVIEW

You will learn more about developing web apps in WebView in the next few chapters. For now, here's an introduction.

Android offers a variety of ways to present content to a user. To provide a user experience that's consistent with the rest of the platform, it's usually best to build a native app that incorporates framework-provided experiences, such as Android App Links or Search. Additionally, you can use Google Play-based experiences, such as App Actions and Slices, where Google Play services is

available. Some apps, however, may need increased control over the UI. In this case, a WebView is a good option for displaying trusted first-party content.

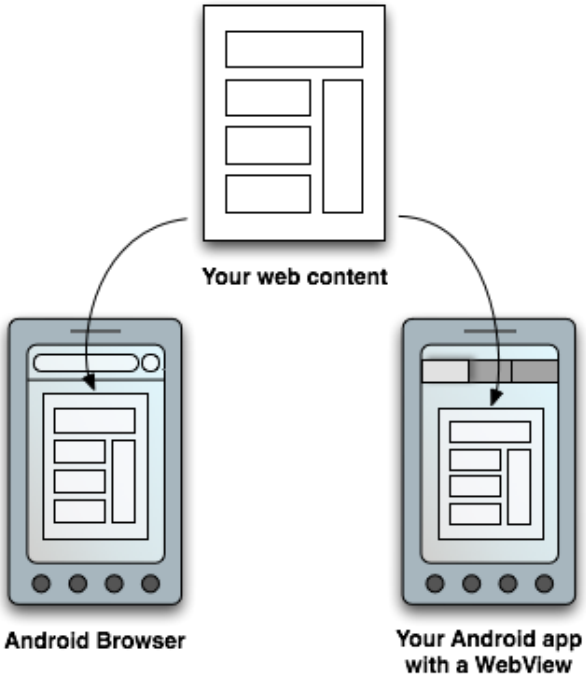


Image from: https://developer.android.com/guide/webapps

The figure above illustrates how you can provide access to your web pages from either a browser or your own Android app. The WebView framework allows you to specify viewport and style properties that make your web pages appear at the proper size and scale on all screen configurations for all major web browsers. You can even define an interface between your Android app and your web pages that allows JavaScript in the web pages to call upon APIs in your app—providing Android APIs to your web-based application.

However, you shouldn't develop an Android app simply as a means to view your website. Rather, the web pages you embed in your app should be designed especially for that environment.

4.1. ALTERNATIVES TO WEBVIEW

Although WebView objects provide increased control over the UI, there are alternatives that may provide similar functionality with various advantages: they require less configuration, may load and perform faster, provide improved privacy protections, and can access the browser's cookies.

Consider using these alternatives to WebView if your app falls into the following use cases:

- If you want to send users to a mobile site, build a progressive web app (PWA).
- If you want to display third-party web content, send an intent to installed web browsers.
- If you want to avoid leaving your app to open the browser, or if you want to customize the browser's UI, use Custom Tabs.

SUPPLEMENTARY LEARNING RESOURCES

Web-based content. (n.d.). Android Developers. Retrieved December 12, 2020 from https://developer.android.com/guide/webapps

SELF-ASSESSMENT QUESTION 4

If you are focusing on building a web app, would you rather build it the traditional way (HTML, Javascript, CSS, etc) or create a mobile app and embed a responsive website using WebView in it? Why?

ENRICHMENT EXERCISE

(30 points) List three titles (app ideas, system title, etc.) that could be developed with the three types of mobile applications. Use the template below:

Title	Best Suitable Application Type	Overview of the Application
Example: Course Schedule Alarm App	Native	Target Users: College Students The application will automatically notify the student with his schedule for the day. (Make this at least 3 sentences.)

REFERENCES

Cover image from: https://buildfire.com/hybrid-vs-native-mobile-app-development-better-data-driven-answer/

Data and file storage overview. (2019). Android Developers. https://developer.android.com/training/data-storage

Mobile Application Development. (2019). Amazon Web Services, Inc. http://aws.amazon.com/mobile/mobile-application-development/

Panhale, M. (2016). *Beginning Hybrid Mobile Application Development*. Apress.

Valdellon, L. (2020). *What are the different types of mobile applications? And how do you choose.* Retrieved from: https://clevertap.com/blog/types-of-mobile-apps/

Web-based content. (n.d.). Android Developers. Retrieved December 12, 2020 from https://developer.android.com/guide/webapps