University of Zakho

MOBILE APPLECATION

Xamarin MiniProject

University of Zakho

By

Helan Ezadeen

Aswan Gahasan

Nazin Jasim

Aisha Rajab

Supervised by

Mr. yousif
Department of Computer Science
2021 – 2022

Contents

Introduction to Xamarin

Xamarin is one of the oldest cross-platform frameworks available. Founded in 2011, Xamarin community has grown to ***1.4M developers*** across 120 countries. The project was acquired by Microsoft in 2016 and became part of its Visual Studio IDE.

The technology is mainly used in enterprise environments and has gained many positive reviews over the years. Xamarin is used by over 15,000 companies in fields like energy, transport, healthcare and others.

What is Xamarin, How, and Where is it Used?

Platforms have become indispensable for application development. And one of the most popular platforms is Xamarin. So, what is it? Read on to learn more about this cross-platform app development tool.

The demand for mobile app development continues to rise in the last few years. Companies have to speed up their software development process to keep up with the rising demand. This is where tools like frameworks and platforms come in. Software development platforms help developers build applications faster and with fewer errors.

One of these tools is Xamarin. The mobile app development platform is one of the most loved by developers, according to Stack Overflow's 2020 survey. So, what is it, and why do developers love using it?

What is Xamarin?

Xamarin is an ***open-source platform*** for building modern and performant applications for ***iOS***, ***Android***, and ***Windows*** with .NET. Xamarin is an abstraction layer that manages communication of shared code with underlying platform code. Xamarin runs in a managed environment that provides conveniences such as memory allocation and garbage collection.

Figure 1. Xamarin — Single Code Base for iOS, Android and Windows Applications

Xamarin applications can be written on ***PC*** or ***Mac*** and compile into native application packages, such as an **.apk** file on Android, or an **.ipa** file on iOS.

C# and XAML

- Xamarin is a platform to create multi-OS apps
- It allows you to create a single application, that can work across **iOS**, **Android** and **Windows** phone.
- In order to develop these applications, you use **C# and XAML.**
- **XAML** acts as the markup and data binding language for the phone app, and **C#** acts as the server side language.

3) How Xamarin Works

Xamarin allows developers to develop native applications for Android, iOS and Windows Phone platforms, with a single codebase, i.e. C# and a single IDE, i.e. Visual Studio . Thus, a developer can be able to develop native mobile applications without knowing Java, Kotlin, Objective-C or Swift. What that means is, all the C# code has to be converted to make it working on these three separate platforms.

Xamarin Promises:

- Xamarin provides native UI interface.
- Xamarin provides native API access.
- Xamarin provides native performance.

Xamarin-Forms Architecture

Xamarin allows you to create native UI on each platform and write business logic in C# that is shared across platforms. In most cases, 80% of application code is sharable using Xamarin.

Xamarin takes care of translating or compiling all your C# code to its corresponding platform-specific code

Xamarin is built on top of **Mono**, an open-source version of the .NET Framework based on the .NET ECMA standards.

Mono has existed for almost as long as the .NET Framework itself, and runs on most platforms including Linux, Unix, FreeBSD, and macOS. The Mono execution environment automatically handles tasks such as memory allocation, garbage collection and interoperability with underlying platforms.

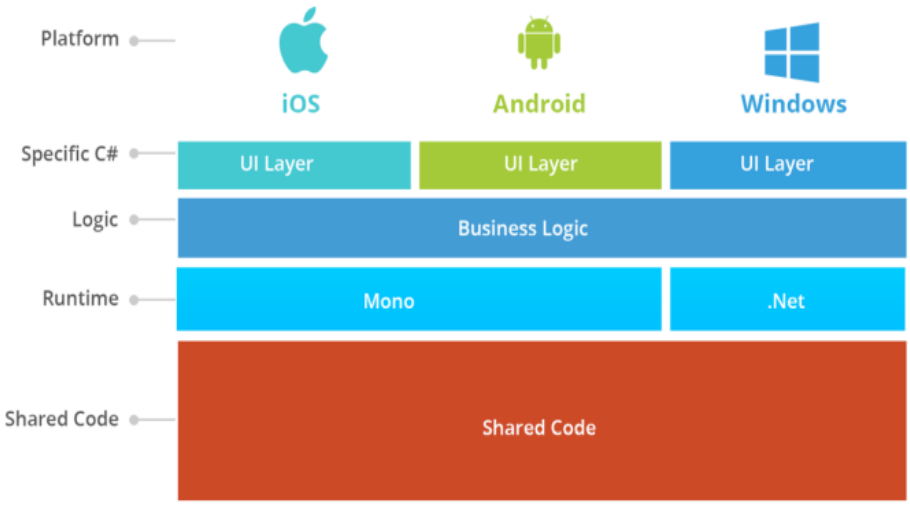


Figure 2. Xamarin Forms Architecture

Xamarin.Android Architecture

For Android, Xamarin leverages the **JIT(Just In Time)** compilation to create an optimized executable.

Xamarin.Android applications run within the **Mono execution environment**. This execution environment runs side-by-side with the **Android Run Time(ART)** virtual machine.

Both runtime environments run on top of the Linux kernel and expose APIs to the code that allows access to the underlying system.

Xamarin.Android applications also contain the **Android Callable Wrappers**(ACW) to allow Android to call into managed code.

Managed Callable Wrappers(MCW) are used whenever managed code needs to call into Android APIs.

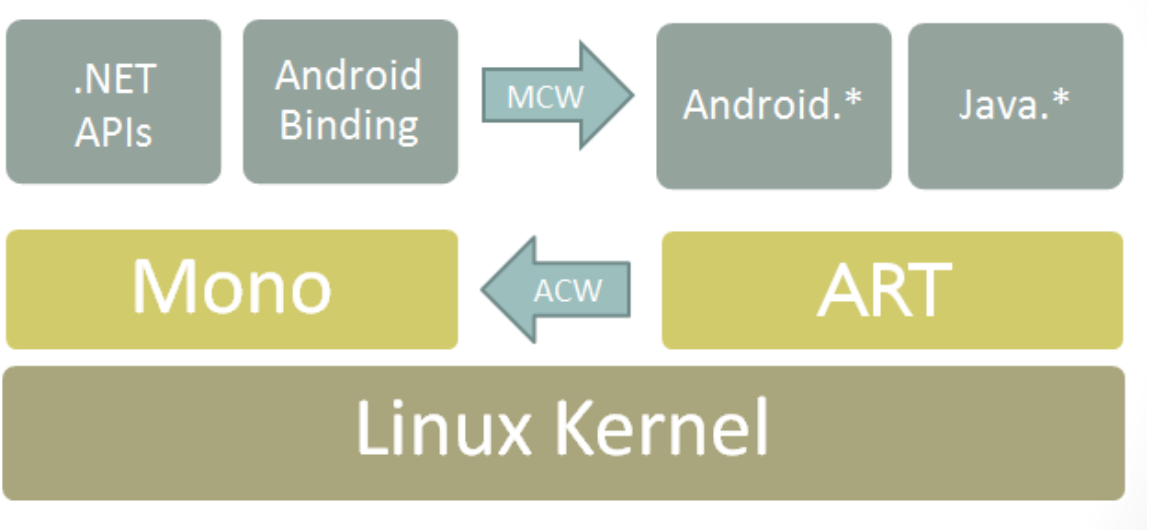
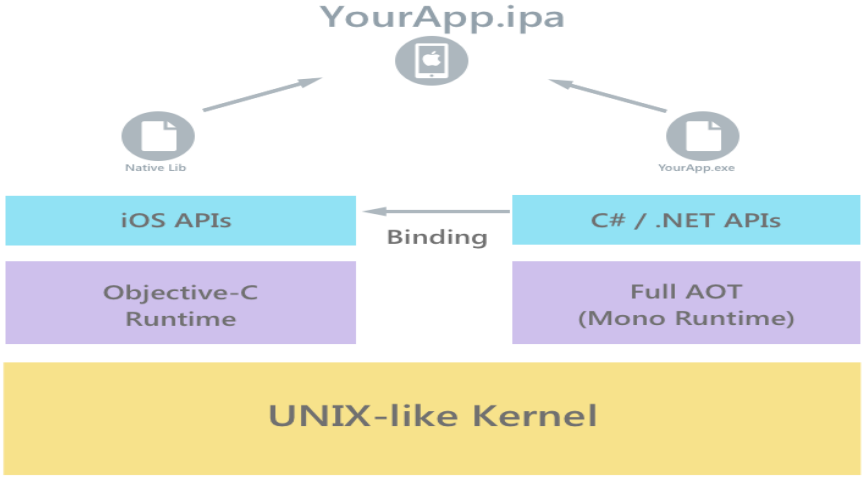


Figure 3. Diagram of Mono and ART above the kernel and below .NET/Java + bindings

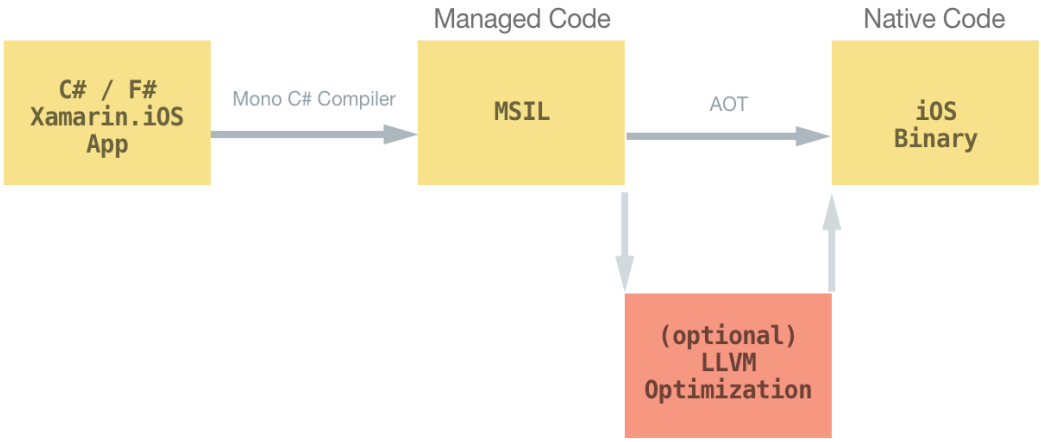
Xamarin.iOS Architecture

For iOS, Xamarin provides a fully compiled **(AOT — Ahead Of Time)** binary that directly runs on your device to provide native performance.

Xamarin.iOS exposes a C#/CIL binding to the Cocoa Touch API and also provides access to ECMA CIL APIs and various other .NET APIs.



Xamarin.iOS runs within the **Mono execution environment** and uses full AOT compilation to compile C# code to assembly language. This runs side-by-side with the **Objective-C runtime**. Both runtime environments run on top of a UNIX-like kernel, specifically XNU, and expose various APIs to the user code allowing developers to access the underlying native or managed system.



Xamarin.iOS instead uses an **Ahead of Time (AOT)** compiler to compile the managed code. This produces a native iOS binary, optionally optimized with LLVM for devices, that can be deployed on Apple's ARM-based processor.

4) System Requirements

MacOS Requirements

- ***Operating System*** **—** macOS Mojave (10.14)
- ***Development Environment*** **—** Visual Studio for Mac
- ***Xamarin.iOS*** **—** Yes and iOS 12 SDK recommended
- ***Xamarin.Android*** **—** Yes and Android 6.0 / API level 23 recommended

- ***Xamarin.Forms* —** (Both iOS and Android) — Xamarin.Forms apps built on macOS can include iOS, Android, and macOS projects, subject to the SDK requirements above. Xamarin.Forms projects for Windows/UWP cannot be built on macOS

- ***Xamarin.Mac* —** *Yes* *and* macOS Mojave (10.14) SDK recommended

Windows Requirements:

- ***Development Environment* —** Visual Studio

- ***Xamarin.iOS* —** Yes (with Mac)

- ***Xamarin.Android* —** *Yes*

- ***Xamarin.Forms* —** Android, Windows/UWP (iOS with Mac computer)

- ***Xamarin.Mac* —** Open project & compile only

5) To be proficient in Xamarin, you need to learn

- **C# language:** It's a very easy language to learn and you don't need to master it anyway. It shouldn't take long.

- **Mobile development**: Start with one (**iOS** or **Android**). You will need to understand how the platform works, how to use controllers, how to store files on the device, how to use all the great features of a smartphone. It's a never-ending path but even getting the basics right takes a long time and is hard.

- **Architecture:** to get the most out of Xamarin, you will want to share as much code as possible and architecture your code &

solution right. It is much harder than having a single application or two different codebase for iOS & Android (even though it's much better in the long term). A good start is deeply understanding POO and patterns such as **MVC** & **MVVM**.

6) The Pros of Choosing Xamarin

Here are some of the biggest advantages to choosing Xamarin:

- **A single tech stack for faster development —** Created with Visual Studio, Xamarin-based apps are developed using a single language: C#. Xamarin apps utilize C# and shared codebases that cover up to 90% of each platform's particular language, APIs and data structure and wrap them in a .NET layer that enables cross-platform development. By developing in C# and allowing Xamarin to handle cross-platform implementations, development teams will be able to accomplish much more with less.

- **Rapid prototyping:** With Xamarin.Forms, developers have access to a complete cross-platform UI toolkit to build interfaces that work on any device. This allows for the creation of a single user interface across all devices, enabling developers to share more code without having to modify the UI for every platform.

- **Native performance and user experience:** In Xamarin it is possible to access each and every native API, so it is possible to use completely native UI, Bluetooth, SDKs etc. Because Xamarin can take full advantage of system- and hardware-specific APIs, apps built using the software will run as well as apps compiled in each platform's preferred language. Users won't be able to tell the

difference between your app and a native app because there isn't one.

- **Native performance and user experience:** In Xamarin it is possible to access each and every native API, so it is possible to use completely native UI, Bluetooth, SDKs etc. Because Xamarin can take full advantage of system- and hardware-specific APIs, apps built using the software will run as well as apps compiled in each platform's preferred language. Users won't be able to tell the difference between your app and a native app because there isn't one.

- **Reduced time to market:** Building apps with shared codebases eliminates time that would typically be spent translating, rewriting or recompiling code to work on different platforms. This shaves weeks, months and possibly years off of the development cycle, allowing apps for all three major platforms to be developed simultaneously. And because these apps are being built together, it means feature parity won't slowly trickle down from your most popular platform to your least — they'll be ready for deployment to all of your platforms at once.

- **Less maintenance:** Maintaining and updating apps built using Xamarin requires less work. Once you've made changes to our source file, they can be applied directly to your apps, eliminating the need to update the source code of your apps individually should any updates, bug fixes or new features become necessary.

- **Apps for all platforms:** What happens if you've got a killer desktop app that needs a mobile version or vice-versa? With Xamarin, it's no problem — developers can create apps for mobile and desktop experiences simultaneously. This also helps

development teams cut back on having to decide whether to develop for just one single platform, as Android, iOS and Windows can be handled simultaneously.

- **Easy to keep updated.** Xamarin takes advantage of native frameworks and usually it takes 1–3 days for iOS and Android platforms to catch up to the latest features. Which is why new platform-specific features can be promptly introduced to your app once Xamarin has been updated.

7) The Cons of Choosing Xamarin

- **It's expensive for enterprises.** Xamarin is free for individuals and small companies, however, enterprises need to purchase a license for Microsoft's Visual Studio. For bare-bones access to Visual Studio without advanced Azure DevOps features or cloud services, single-user licenses start at $499. Enterprise users requiring all the bells and whistles pay up to $2,999 for an annual subscription to Visual Studio Enterprise.

- **Complicated to use all open-source libraries.** While Xamarin does support most of .Net libraries, it doesn't support all of the available 3rd-party libraries for Android and iOS without specific wrappers.

- **Not suitable for apps with heavy graphics.** Each platform has a different method for visually laying out screens.If the application has rich UX/UI, it should be implemented natively.

- **Larger App Size**. Xamarin adds 3–5 megabytes for the release and around 20 megabytes for debug builds.

8) Xamarin Comparison with other Cross platforms(Xamarin vs Flutter vs React NativeApp)

Speed:

App speed is always a concern for the app owners so it has to be quick and responsive. Flutter offers faster app development with faster app speed at the output. Other platforms like React Native and Xamarin also performs better in the app speed but whilst comparing with the Flutter they are slightly slower. The app speed also depends on the development process, functions, features, and other terms so it's hard to compare on the instant but we can conclude the **Flutter** as a **faster cross-platform** app development framework.

Community Support:

Community support is built to give support and point out any issues related to the framework. While talking about community support, Xamarin lost in the competition because it has limited community support. On the other hand, React Native and Flutter have a better community for solving any issues and make it better for the users. According to the GitHub, React Native has more stars and Followers compare to the Flutter. So we can say **React Native** is the clear winner in terms of community supports.

Security:

Security of the source code and application is a must thing that's why security is listed in the priority action for all cross-platform developers. As mentioned earlier, React Native support third-party plugins and not have robust security supports. React Native lose the race of security, so let's talk about other cross

platforms. **Flutter** and **Xamarin** based apps are robust and compatible with security access.

Customization:

As we know the React Native allows third-party plugins that simply means customization with the React Native is easy and compatible with Flux. **Flutter's Hot Reload feature** and **Xamarin Live Reload** feature empower both platform for live changes in the on-going development. Flutter and Xamarin have pre-install layout elements that simply indicates the freedom of customization with both cross-platform app development.

Usability By Developers:

The Xamarin is based on the C# that is simple and easy to learn the language. Flutter works on the Dart, an object oriented programming which is also easy to learn for newbie developers. React Native coded in Java that is most widely used in any development so working on Java is a piece of cake for every developer. In the concise, **Xamarin** and **React Native** are the easiest to access cross-platform app development framework.

Popularity:

According to **Xamarin**, the cross-platform tool is used by over 1.4 million **developers** worldwide. Moreover, **Xamarin** products are used by over 15,000 companies from over 120 countries in different industries such as media, transportation, finance, healthcare, and gaming. Due to its ability to write native UI code for app development, many popular brands use the

Xamarin for crafting their application. Xamarin develops native codes but it is not free, so it is considered for the premium users. As most of the developers use Xamarin but still many developers rely on the Flutter and React Native.

Popular Apps that used React Native, Xamarin and Flutter

React Native Platform

- Walmart
- SoundCloud
- Bloomberg
- FaceBook
- Instagram

Xamarin Platform

- Storyo
- SuperGiant Games
- The World Bank
- APX
- Skulls of the Shogun

Flutter Platform

- Alibaba
- Hamilton Musical

- Google AdWords
- AppTree
- Google Greentea

9)Why Choose Xamarin for App Development over React Native & Flutter?

Xamarin is equipped with robust emulators that suit various mobile platforms, and this is an important reason why many business organizations are relying on this cross-platform app development framework. Xamarin provides Several options for debugging with the freedom to crosscheck from the desktop, emulator, or directly on the device. Here are some of the noteworthy features of this framework that developers are enjoying.

- **Performance:** The performance level of apps is highly competitive compared to hybrid or any other cross-platform development tools. Image loading is 14% faster when compared to other platforms, and image-saving speed is super-fast as well.
- **Development Speed:** Xamarin comes with a library of templates that permits to use of standard interface elements. Developers enjoy when the inception to development speed is faster with Xamarin.
- **Sharable Code:** Developers are equipped with the feature to write C# code that can run on cross-platforms. PCL (Portable Class Libraries), Shared Projects, .Net Standard Libraries facilitate code sharing. Adding more value to this feature, the Xamarin.forms framework makes it possible to share the same code on several other platforms.

- **Native UX:** It is compatible with different operating systems and enables developers to achieve native look whether they develop the app for Android or iOS.
- **Open Source:** Xamarin is an open-source free tool that comes with Microsoft's development environment Visual Studio. It gives you the freedom to use built-in tools for manual customization along with an option to reuse the codes.
- **Resourceful:** Developers can go in-depth if they want to understand how exactly the platform works. Xamarin is owned by Microsoft and the platform is focused on simplifying the cross-platform app development process.

Xamarin comes with a compilation of tools like Xamarin.mac, Xamarin.insights and Xamarin.testcloud.

It gives you full freedom to create Mac Apps, analyze apps through Insights and run automated tests with the Xamarin test cloud.

10) Why Xamarin Is a Win?

If you're a developer showing any inkling of needing apps developed for multiple platforms, Xamarin is one of the best tools available to make your life easier.

Xamarin comes with a range of rich features. Coupled with an ever-growing community of developers, it has become a personal favourite of experienced

and new developers alike when it comes to using a cross-platform development framework.

Thanks to Xamarin's powerful C# environment, native and cross-platform libraries and APIs, and ease of deployment, it's the best choice to keep Android, iOS and Windows apps developed in sync. This, in turn, reduces the overall time of development and brings new features to your users faster.

The approach is as simple as write once run anywhere. You can skip the development stages during tight deadlines.The community is expected to receive a lot of application updates from Xamarin.

Enterprises can adopt this framework because it doesn't require native iOS and Andriod platforms.This tool can extend the app development to the next level and is easily accessible.

Xamarin is designed to scale up and is open-source, and for many businesses, this cross-platform framework has become an obvious choice.

Features

What is Xamarin's strong point? It allows for sharing 90 percent of your code to major platforms. Aside from that, the cross-development platform also offers features that many developers love, such as:

1. Software Development Kits (SDK) Binding

Xamarin has the bindings for all the platform SDKs for Android and iOS. Furthermore, these bindings are easy to use and navigate. They also provide robust compile-time type checking. In short, these bindings can help in developing more error-free and higher-quality applications.

2. Wide Arrays of Third-Party Codes

The platform provides you with facilities to apply Java, Objective-C, and C++ libraries directly. Thus, allowing you to use wide arrays of third-party codes. In addition, Xamarin has project binding capabilities that let you tie Java libraries and native Objective-C by using declarative syntax.

3. Use of Modern Language Constructs

All Xamarin applications are developed in the C# programming language. It is a modern language that features more dynamic functional constructs like parallel programming, lambdas, LINQ, and more.

4. Cross-Platform Support for Mobile App Development

It offers cross-platform support for Android, Windows, and iOS. This means you can share 90% of your codes to any or all of the three platforms. Aside from this, using the tool will also allow you to access common resources through a unified API across all platforms. Thus, you can significantly reduce the development time and cost.

Who Is It For?

With these features, the open-source platform is perfect for developers who are:

- Planning to write, share, and test codes and business logic across different platforms.
- Developing cross-platform applications using C# and Visual Studio.

Provides Native Experience

Most cross-platform development tools strip their platforms of their uniqueness. Xamarin, on the other hand, highlights the platforms' strengths. It does this by converting Objective-C, Java, and .NET to C#.

Developers would then be able to build software that fits all operating systems using the shared codebase and libraries. As a result, the software provides the complete user native experience.

Covers Most Mobile Operating System

Other tools allow development for Android and iOS systems; the instrument takes it a step further. It covers operating systems of Windows Phone and Blackberry.

Reduces Cost

Like all cross-platform development tools, it eliminates the need to develop separate apps for different operating systems. So, there is no need to employ additional developers to create apps for other operating systems.

In addition, you can reduce the maintenance costs as a single team can do the troubleshooting after deployment.

Quicker Development to Market-Time and Lower Maintenance Downtime

Because the tool is cross-platform, you only need to write the code once. Then, you can share business logic across all mobile operating systems. Thus, hastening the software development process, which allows you to release your product quickly.

Moreover, downtime due to maintenance and troubleshooting will be shorter. Your team will only need to check and fix issues in one operating system and share it with the rest.

It's an Open-Source Platform

The cross-platform mobile app development tool is part of Microsoft's open-source .NET platform. This means that it is free and has strong community support. Over 3,700 companies are contributing to enhancing and forward the development platform.

As stated above, Xamarin offers developers and companies alike many benefits. Its features help in reducing both the software development time and cost. So, where can you find the best Xamarin developers to help you with your mobile app development projects?

Where to Find Xamarin Developers?

The first step in finding the best Xamarin developers is to detail their responsibilities first. These developers play critical roles in the design and development of cross-platform mobile services and solutions.

Therefore, they must have experience in mobile development for iOS, Android, and Windows. Plus, they must also have excellent communication skills to work with your team seamlessly. However, with the rising need for IT professionals in the US, you will encounter numerous competitions in recruiting the best Xamarin developers.

Luckily, another option available for you is to hire offshore software developers. And, that is where we at Full Scale come in. All of our Xamarin developers have undergone a rigorous recruitment and onboarding process.

Aside from that, these developers and other experts are continuously training to be up-to-date with the latest technologies and techniques. Hence, our experienced and professional developers, programmers, testers, and other specialists are top of their fields.

With us, you can build your dream software development team to fulfill your mobile application projects.

Modules Description

In this project, we have Two modules

1) Data gathering and pre-processing.

2) Applying an Algorithm for prediction.

Explanation:

1) In this module we first gather the data(database) for our prediction model. Data comes in all forms, most of it being very messy and unstructured. They rarely come ready to use. Databases, large and small, come with a variety of

issues- invalid fields, missing and additional values, and values that are in forms different from the ones we require. To bring it to workable or structured form, we need to "clean" our data, and make it ready to use. Some common cleaning includes parsing, converting to one-hot, removing unnecessary data, etc.

In our case, our data has some days where some factors weren't recorded. And the rainfall in cm was marked as T if there was trace precipitation. Our algorithm requires numbers, so we can't work with alphabets popping up in our data. so we need to clean the data before applying it to our model.

2)Once the data is cleaned, In this module that cleaned data can be used as an input to our Linear regression model. Linear regression is a linear approach to form a relationship between a dependent variable and many independent explanatory variables. This is done by plotting a line that fits our scatter plot the best, ie, with the least errors. This gives value predictions, ie, how much, by substituting the independent values in the line equation.

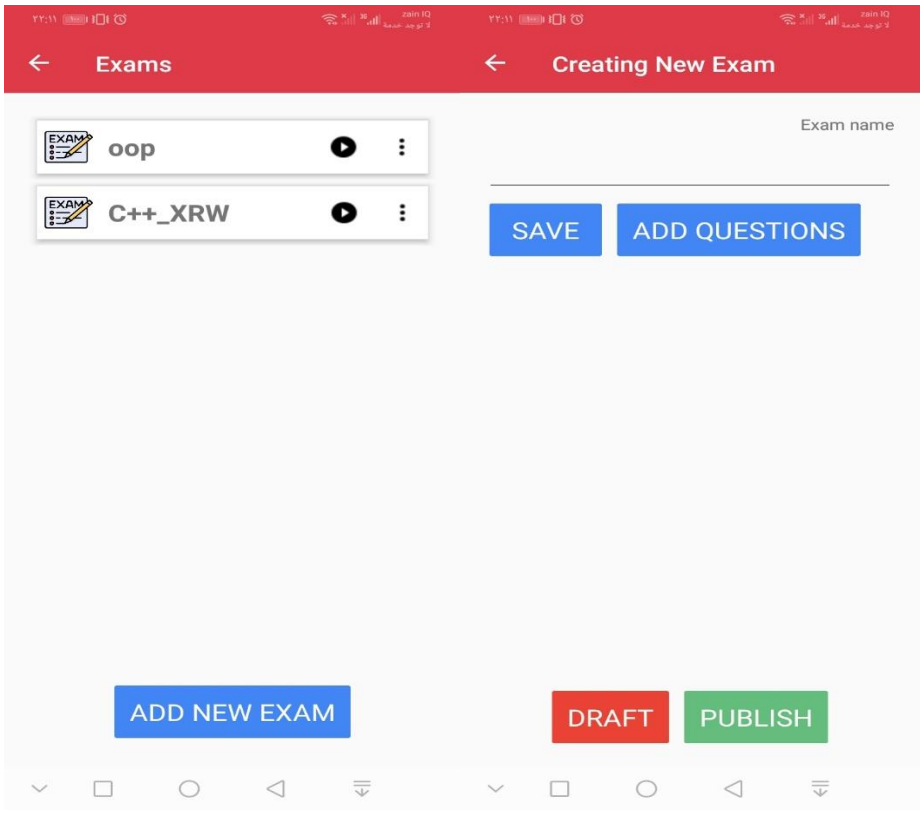
We will use Xamarin model to train our dataset. Once the model is trained, we can give our inputs for the various columns such as temperature, dew point, pressure, etc. to predict the weather based on these attributes.

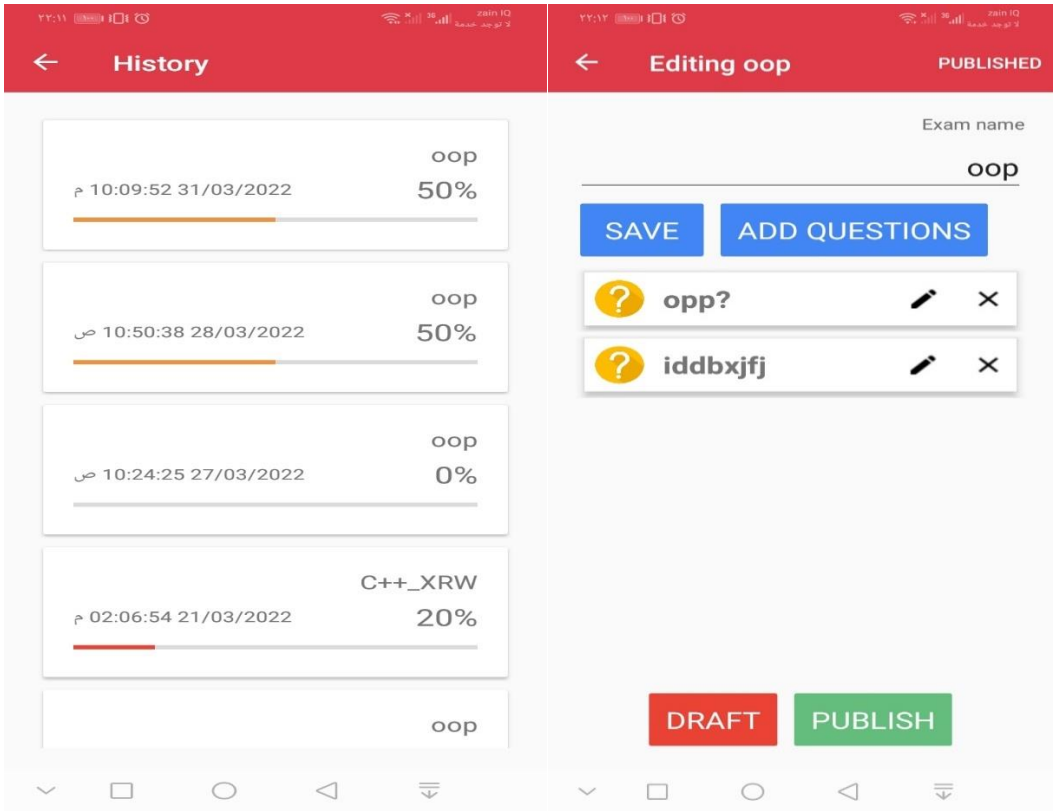
Module Outcomes:

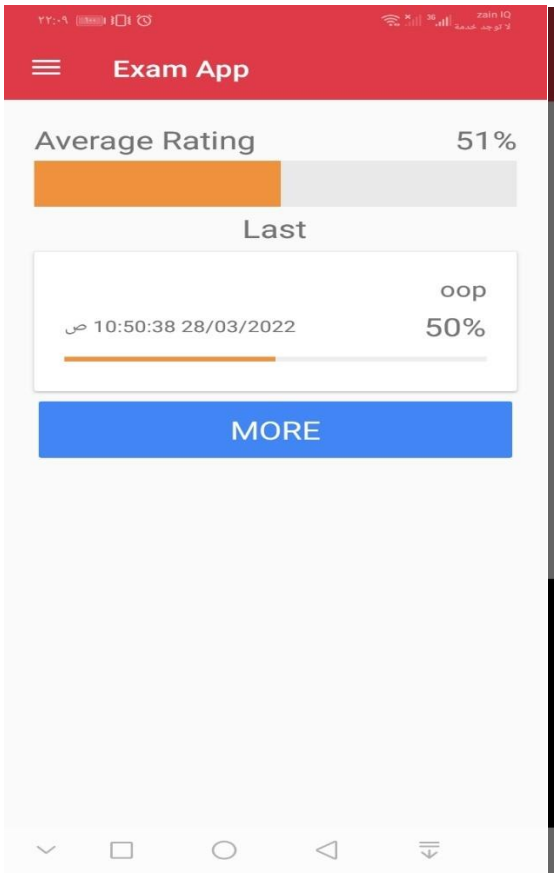
1) By the end of the first module the fully cleaned and useful data is available for the apply the algorithm for the prediction.

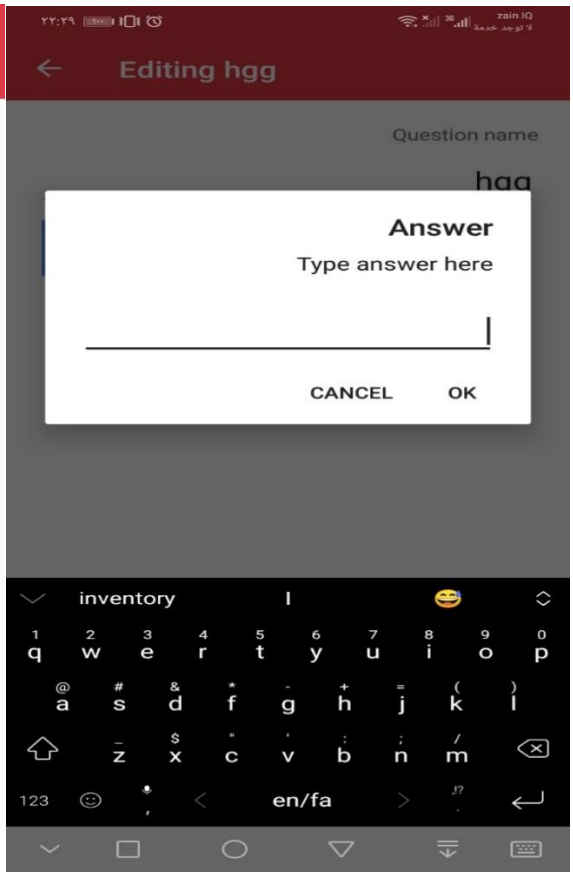
1) By the end of the second module the actual prediction will happen the outcome is the amount of rainfall in inches based upon the user's input.

Project Pictures






Exam App
Average Rating
51%
Last
oop
10:50:38 28/03/2022 ص
50%
MORE


Editing hgg
Question name
hgg
Answer
Type answer here
CANCEL
OK
inventory
en/fa


Editing opp?
Question name
opp?
SAVE
ADD ANSWER
object ori...
oriented o...
NEW QUESTION

3. CONCLUSION

In this small project, we completed the two exams program with choices, regarding questions and answers, the right choice within the choices, adding several topics and questions about these topics, the test result and the final result rate.

4. REFERENCES

Textbooks:-

i. ^ "Xamarin delivers tool for building native Mac OS X apps with C#". December 13, 2012. Archived from the original on April 7, 2014. Retrieved April 1, 2014.
ii. ^ "Xamarin for Android". Archived from the original on April 23, 2014. Retrieved April 1, 2014.
iii. ^ "Xamarin for iOS". Archived from the original on March 30, 2014. Retrieved April 1, 2014.
iv. ^ Peter Bright (February 20, 2013). "Xamarin 2.0 reviewed: iOS development comes to Visual Studio". Archived from the original on April 14, 2014. Retrieved April 1, 2014.
v. ^ Mikael Ricknäs (June 25, 2013). "Xamarin tool aims to show the ease with which .NET apps can become mobile". Archived from the original on April 7, 2014. Retrieved April 1, 2014.
vi. ^ "Announcing Xamarin 3".
vii. ^ "Windows Platform Features - Xamarin". docs.microsoft.com.

viii. ^ Krill, Paul (January 14, 2020). "Microsoft enables native mobile development with Blazor". InfoWorld. Retrieved February 6, 2020.
ix. ^ Billson, Alex (July 15, 2018). "Cross Platform Mobile Apps with .NET and Uno". Hacker Noon. Retrieved January 20, 2019.
x. ^ "Xamarin Updates From Microsoft Build 2020". Xamarin Blog. May 19, 2020. Retrieved May 28, 2020.
xi. ^ "Introducing .NET Multi-platform App UI". .NET Blog. May 19, 2020. Retrieved June 4, 2021.
xii. ^ "dotnet/maui". GitHub. Retrieved May 28, 2020.
xiii. ^ "Xamarin Test Cloud". Archived from the original on April 7, 2014. Retrieved April 1, 2014.