Comparative Analysis of Hybrid Mobile App Development Frameworks

Mohit Singh, Shobha G

***Abstract:** With the rise of mobile devices and their usage, a lot of development has been made in terms of the development of applications for mobile devices. Traditionally, app development was restricted to the particular operating system, and a separate codebase was required for applications to be developed for multiple operating systems. A new paradigm of development took place in recent years which was of Hybrid app development, leading to the development of multiple frameworks which allowed for a single codebase to be used for multiple operating systems. This paper explores the features and analysis of different hybrid app development frameworks available in the market. A comprehensive analysis has been made to compare the different frameworks which are cross-platform and support web, Android, and iOS platforms. The analysis shows that all the frameworks have their merits and usage of anyone framework over others can vary from case-to-case basis. The detailed analysis of the features will bring a general conclusion over the choice of framework.*

***Keywords:** Hybrid App Development, Cross-Platform Apps, Hybrid App Frameworks*

I. INTRODUCTION

There are over 3 billion mobile devices that are currently in use [1]. The major platforms on which these devices work is Android-based, owned by Google, iOS owned by Apple, and a few Windows phones [2]. The development of applications for these platforms becomes a challenging task when we consider building for all the platforms out there, with each platform having its own set of tools and libraries and even different programming languages to work upon. The developer if they want to launch their application to multiple platforms which are vendor-specific such as iOS, and Google-based Android, has to learn a lot of things such as programming languages, tools, and libraries for each of them. [3] This arises a lot of confusion among developers who are starting on which platform to work, and where to build their apps, considering the popularity of Android and iOS, both seem to be viable and great marketplaces for the developers. So, the developers have a choice to either go for native application development, which will be specific to a particular operating system, a web app based on HTML and related technologies which would work on all platforms but will not provide the benefits and features available to native one [4].

Manuscript received on July 13, 2021.
Revised Manuscript received on July 16, 2021.
Manuscript published on July 30, 2021.
* Correspondence Author

Mohit Singh*, Department of Computer Science & Engineering, RV College of Engineering, Bengaluru, India. Email: mohitsingh.cs17@rvce.edu.in

Shobha G, Department of Computer Science & Engineering, RV College of Engineering, Bengaluru, India. Email: shobhag@rvce.edu.in

The third option is hybrid app development, which seems a lucrative option, where there will be an option to use a single codebase for multiple platforms.

Hybrid app development ensures cross-platform application development, the major of which is Android and iOS compatibility. We explore different hybrid app development frameworks available today which have large community support and tools available. There are several hybrid app development frameworks available out of which a few of them have gained popularity and are being used by a large number of developers. [5]

The frameworks can be differentiated from each other on multiple factors such as programming language, and support of operating systems. Considering the support of operating systems to be a major factor in choosing the framework, we look into the different frameworks available to us [6]. React native is a widely used framework available with support for web, Android, and iOS platforms. [7] Other platforms which have support for web, Android, and iOS operating systems include the Ionic framework, NativeScript, and Flutter. Xamarin is another hybrid app development framework available that has support for Windows along with Android and iOS and uses the .NET framework beneath it. [6]

The comparison of these frameworks becomes essential from business and development use cases and can be crucial for a business unit when deciding on product development. The main advantage of using hybrid app development frameworks can lead to a great reduction in time and cost for a business unit, as these platforms including ReactNative, Ionic, and NativeScript are web-based and have a marginal cost compared to the development of native apps. [8] So, the choice of a hybrid app framework is important for a company to make an informed decision.

The paper aims to compare the hybrid app frameworks and allow a generalized conclusion over the framework which could be used for development depending on the different features and factors which differentiate the different frameworks from each other. The structure of the paper is divided as – Section 2 describes the comparison between native and hybrid apps which discusses the need for hybrid apps and some drawbacks of the native way. Section 3 gives the architecture of hybrid app frameworks and we have tried to abstract a general architecture used by almost all hybrid frameworks. Section 4 is our research methodology which lists out the choice of framework, different tools and IDEs available for development, native features implementation in hybrid framework, code reusability, and testing. Section 5 concludes the paper with the further need of research areas and alternatives available for hybrid app frameworks.

II. NATIVE VS HYBRID APP DEVELOPMENT

Native mobile applications are those which are built by using the specific programming language for which it is designed for. An example would be using Java and Kotlin for developing native Android applications and using Swift as the programming language for the development of iOS applications. [9] The native development has its own merits, such as it provides its dedicated tools, library, and integrated development environment (IDEs) for development. Android provides Android Studio, which is managed by Google, and to develop iOS apps with Swift, we need XCode as the IDE, provided by Apple. These IDEs are a complete set of software that includes editor, compiler, and debugger, along with layout designer and integrated version control support [10]. These features reduce the development time considerably and increase the efficiency of developers by allowing them to debug and fix bugs easily.

Native mobile applications offer several advantages some of which have been discussed below-

- Native applications are faster to execute as they have been written in the native language and can make use of device hardware efficiently.
- Unlike web apps which mostly require network connectivity to work, native apps can work offline without any internet support and resume the connection when the network comes back. This is taken care of by caching, and local database support for native apps.
- The elements and layout of the application are tightly coupled with the look and feel of the operating system they are built for. This is because most of these elements have been designed by the owner of operating systems.
- Native apps support a larger number of hardware sensors such as GPS, Bluetooth, and Camera and offer better control for using these features [14].

Despite having a plethora of benefits associated with them, if compared to hybrid applications they have some demerits which have given rise to hybrid apps. Some of the limitations associated with native apps are given below-

- The lack of flexibility of platforms is a major issue in native apps. Developers tend to restrict to a single platform as the learning curve associated is very high for each platform.
- The development of native apps is a costly process if the business or a company wants to launch its application for multiple platforms. They need to have separate teams for development for various platforms which they are targeting. The maintenance costs are also high for native apps.
- As each platform will require different codebases, the time taken for development and bringing the product to market can be much higher as compared to the hybrid app model.
- Bug fixes and upgrades need to be done through the different app stores which can take time, and is a slow and costly affair to maintain the app.

These factors gave rise to usage and increased demand for hybrid apps. Hybrid app frameworks mostly use web components beneath them and cross-platform app development came to increased usage only after HTML5 came in development scenario which opened a lot of options. [9] The hybrid apps are platforms that allow a single codebase to be used for multiple platforms and can be said to follow the principle of 'write once and run anywhere. The majority of hybrid app frameworks use HTML5, CSS, and JavaScript beneath them. This code is then translated to the operating system on which they are supposed to run. It requires minimal or no changes in the codebase. The main advantages of using a hybrid app framework to develop your applications are listed below-

- Hybrid apps can offer rich UI components which have been taken from web counterparts and take benefit of both native and hybrid elements.
- Owing to a single codebase to be used by multiple platforms, they offer a wider reach of audience.
- The development time can be considerably reduced as only one codebase has to be developed for multiple platforms.
- The maintainability of code is easier compared to native apps.
- The cost of development is widely reduced as the time to development and codebase is reduced.

III. HYBRID APP FRAMEWORK ARCHITECTURE

Hybrid apps are usually built on HTML5, CSS, and JavaScript-based framework, and mostly encompass web components that make up the UI of the application, which then use plugins or libraries to interact with operating system-specific features such as Camera, Geolocation, and other hardware sensors [9]. These libraries interact with operating system APIs which might be in the native language of the platform such as Java for Android. However, the hybrid app is concerned with the usage of the APIs and not the underlying code or architecture beneath it, and therefore can be abstracted to make use of those libraries.

Interpreted hybrid framework approach refers to using a common language usually JavaScript for writing all the user interface elements and then generating native elements from these. [9] The native APIs are provided by abstracting them from system OS APIs and using them in the application.

Fig. 1 is an architecture for the Ionic framework which is a hybrid app framework widely used in the mobile platform, to develop apps for web, Android, and iOS simultaneously using a single code base.

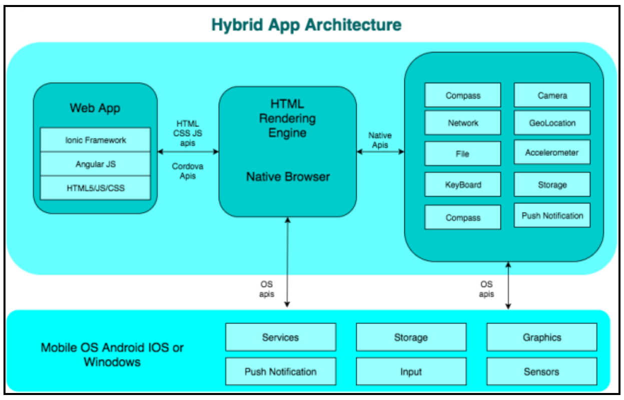


Fig. 1.Ionic Framework app architecture.

The application logic is written in form of a web app using HTML, CSS, and Angular which is a JavaScript-based framework for writing web apps. The framework uses Cordova and Capacitor for interfacing with native APIs, to use the native features of the app such as Compass, Camera, Geolocation, and Push Notifications among others. [11] The app uses OS APIs to interact with these hardware and native features.

IV. RESEARCH METHODOLOGY

We compare and analyze different hybrid app frameworks which are popular in use and show trends of increasing usage over time. These frameworks include React Native which has been developed by Facebook, Ionic Framework developed by Ionic, Flutter developed by Google, and Xamarin by Microsoft. The applications have been compared analytically and a technical approach to compare the frameworks is not possible due to the varied nature of the framework and different use cases they serve to. However, we have tried to compare them across some parameters which are common for a developer and a business to analyze their choice of framework for their application development task.

A. *Choice of Framework*

The choice of framework to be included in this paper was largely dependent upon its prevalent usage and trends as given in fig 2 and its rising usage over the years. These frameworks have been the market leaders in the hybrid app development domain, and are in a continuous phase to do so, for the upcoming years. [12] So comparing them was an obvious choice. There are other hybrid app frameworks available including Quasar, KendoUI, Framework7, Aurelia, OnsenUI, and ExtJS to name a few, but the overall usage of these frameworks is negligible as compared to the other ones.

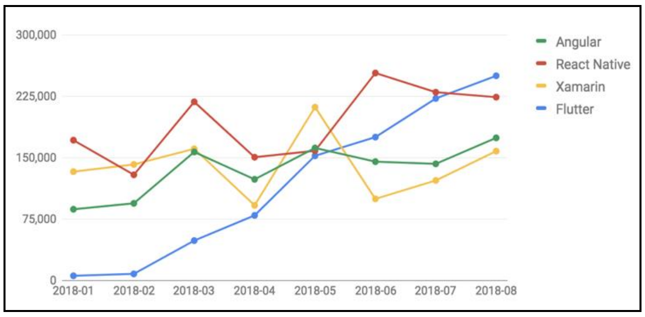


Fig. 2.Usage of various Hybrid App Frameworks.

B. *Programming Language Tools and IDEs*

Let us look at the frameworks one by one by comparing the language which they are built-in for the development of apps, and the IDEs which they support or in which the code has to be written for them.

React Native developed by Facebook, uses Javascript as its primary language for writing the code for the apps in it and uses web-based tools such as HTML and CSS to design the app layout and code. [7] The popularity of React Native arises from its ability to mimic the native feel of the app. It does so by using native components internally instead of web components. It is a javascript-based framework and hence can be written in any IDE, out of which Visual Studio Code is a preferred one.

The ionic framework has been developed by Drifty and is. Cross-platform development framework which uses Angular framework is a Javascript-based framework for development. Ionic uses web-based technologies such as HTML and CSS to build UI elements. However, it also provides its Ionic components which can be extremely helpful in responsively designing apps, for different screen sizes and resolutions [11]. It uses Cordova and Capacitor which are tools developed by Ionic to interact with native mobile elements and for using the hardware features of the mobile operating system. The IDE preferred to be used with Ionic is Visual Studio Code.

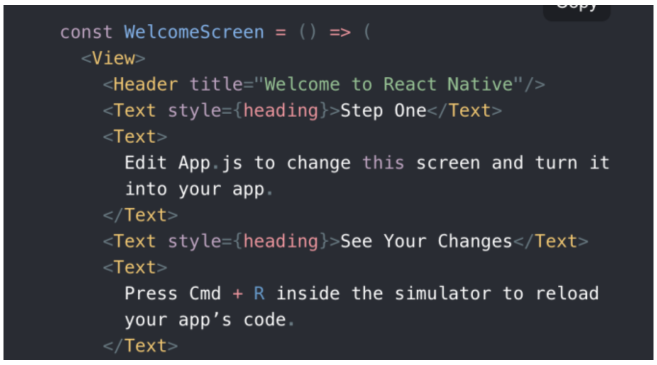
Flutter is a hybrid UI framework that has been developed by Google, and it uses a new programming language known as Dart to write its code on. Dart is an open-source language that translates the elements to the native look and feel of the operating system for which the app is being built. Flutter components compile directly to native feel, which is different from [13] Ionic which uses web components, and React Native which offers native components to be used. Since it is directly compiled to native, it offers faster performance. It also uses VS Code for writing the application.

Xamarin is a cross-platform framework, which is open source and developed by Microsoft. It uses C# to write the cross-platform code which is later compiled to other operating systems to native code for that specific platform. It uses native UI elements and takes a very different approach to development, unlike other hybrid framework approaches which use web-based components [15]. Xamarin uses Visual Studio as its IDE for writing the application code, which is free for community use but costs for using it in enterprise mode, which can be huge for a small business.

C. *UI Development Tools*

UI development is a major task while developing mobile applications and each hybrid framework caters to UI development in its way, which differs from the language in which UI elements are written to the way they are rendered.

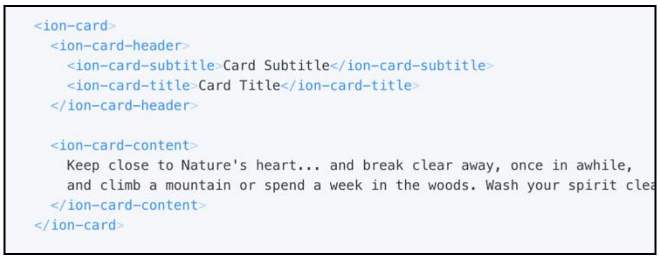
React Native UI elements are written in HTML and styled using CSS. It uses a mixture of JavaScript and a custom markup language called JSX for writing the UI code [7]. An example of JSX code is given in fig 3.



```
const WelcomeScreen = () => (
  <View>
    <Header title="Welcome to React Native"/>
    <Text style={heading}>Step One</Text>
    <Text>
      Edit App.js to change this screen and turn it
      into your app.
    </Text>
    <Text style={heading}>See Your Changes</Text>
    <Text>
      Press Cmd + R inside the simulator to reload
      your app’s code.
    </Text>
```

Fig. 3.JSX Markup Code for React Native UI.

The ionic framework has its UI component written in HTML and CSS. Apart from this, Ionic provides its own set of UI components which can ease the development of UI to a great extent through the use of grid layout-based design and using Ionic components such as cards, tables among others which have been developed by the Ionic team and are available to use. An example of the Ionic card component is given in fig 4.

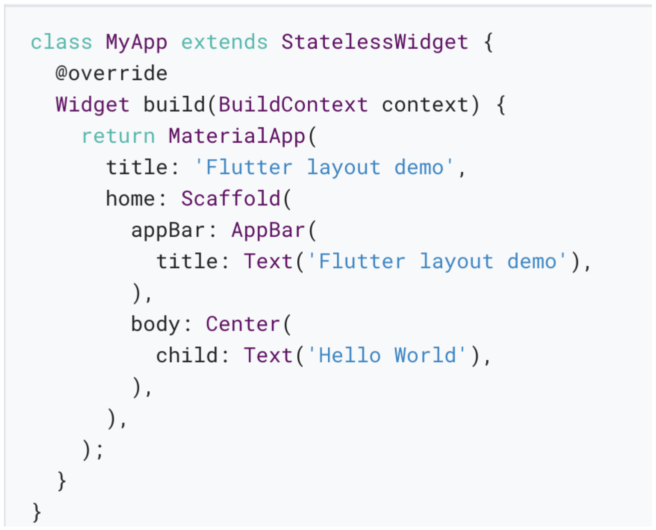


```
<ion-card>
  <ion-card-header>
    <ion-card-subtitle>Card Subtitle</ion-card-subtitle>
    <ion-card-title>Card Title</ion-card-title>
  </ion-card-header>

  <ion-card-content>
    Keep close to Nature's heart... and break clear away, once in awhile,
    and climb a mountain or spend a week in the woods. Wash your spirit clea
  </ion-card-content>
</ion-card>
```

Fig. 4.Ionic Card Component in use.

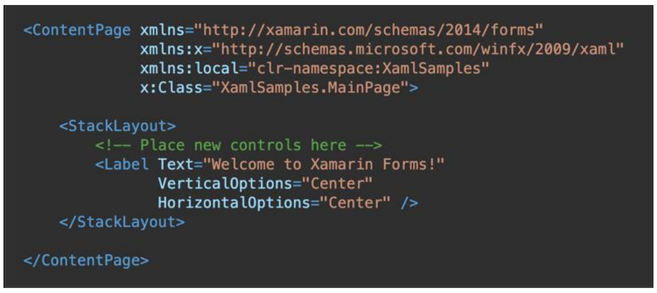
Flutter makes use of the Dart programming language for writing the layouts and UI components of the application. It lists all its UI elements as widgets that are inspired by React's view of native components, and widgets are classes used to build UIs and layouts. A sample code of flutter widgets is given in fig 5.



```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter layout demo',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Flutter layout demo'),
        ),
        body: Center(
          child: Text('Hello World'),
        ),
      ),
    );
  }
}
```

Fig. 5.Flutter layout and component using Widgets.

Xamarin uses XAML for the development of its UI elements which are a part of Xamarin forms-based architectural layout [15]. These elements are then compiled into native elements when the application is compiled to be used on different operating system platforms. An example of Xamarin XAML layout is given in fig 6.



```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:XamlSamples"
             x:Class="XamlSamples.MainPage">

    <StackLayout>
        <!-- Place new controls here -->
        <Label Text="Welcome to Xamarin Forms!"
               VerticalOptions="Center"
               HorizontalOptions="Center" />
    </StackLayout>

</ContentPage>
```

Fig. 6.Xamarin XAML Layout example.

D. Operating Systems Supported

React Native supports both Android and iOS operating systems, by using Native APIs which are platform-specific for handling the interaction between react-native components and operating systems.

The ionic framework supports Web, Android, and iOS through the use of a single codebase. The support of native elements is done by Cordova and Capacitor which are tools to bridge the gap between web components to native OS APIs and allow the use of Camera, Geolocation, among others to Ionic apps.

Flutter supports Android and iOS operating systems. The code is written in Dart and is compiled to native languages which can be used in the two operating systems mentioned above.

Xamarin supports Android, iOS, and Windows 10 Universal Windows Platform. The code is written in C#, which is compiled to native language elements and then used.

E. Code Reusability

Most of the code written in hybrid app frameworks is reusable across different platforms, however, to access device hardware features like camera, GPS, contacts, either native language code has to be written, or the framework is smart enough to compile the single codebase to native elements of that platform. The different hybrid application frameworks listed here perform differently in terms of code reusability.

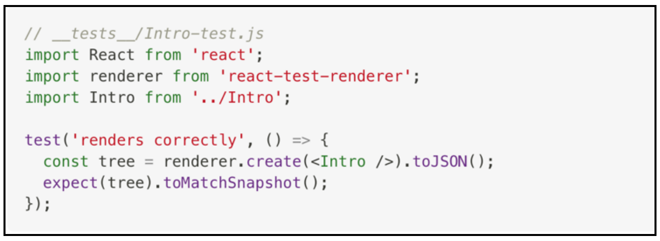
React Native has up to 90% of code reusability support, Ionic uses web components and is not compiled into native, so almost 98% of the code is reusable, Xamarin too since it uses its XAML for designing app components, 98% of Xamarin code is reusable across different platforms. Flutter lacks behind in terms of code reusability with only 50-90% of the code being reusable [5].

F. Testing

Testing is an integral component of app development and makes a crucial part of a robust and error-free application. In native app development platforms, tools like Espresso and JUnit are available for UI testing and code testing [16]. Similarly, for iOS, OCUnit and UI Automation tools are available for testing. A good framework is expected to have a robust testing tool and we look at the testing solutions available for various hybrid app frameworks.

For React Native, testing tools available for JavaScript can be used, and therefore for testing, tools like ESLint are available for static analysis, and for unit testing, frameworks such as JestJS can be used. A sample example is provided in fig 7 which tests an HTML render test.

Ionic is based on Angular, which is a JavaScript-based framework that can also use the same tools which React Native uses, such as EsLint, and Jest framework for unit testing.

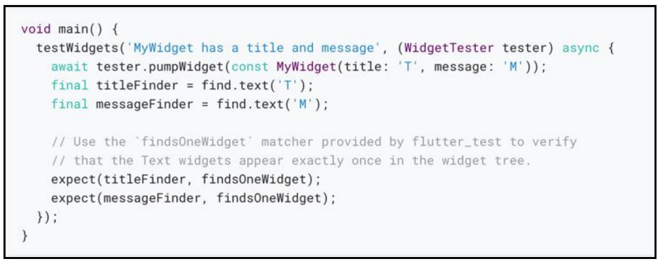


```
// __tests__/Intro-test.js
import React from 'react';
import renderer from 'react-test-renderer';
import Intro from '../Intro';

test('renders correctly', () => {
  const tree = renderer.create(<Intro />).toJSON();
  expect(tree).toMatchSnapshot();
});
```

Fig. 7.JestJs testing example in React Native

Flutter apps can be tested by the test package which is provided by Flutter itself, to cover unit tests for widgets. An example of such a test for flutter widgets is given in fig 8.

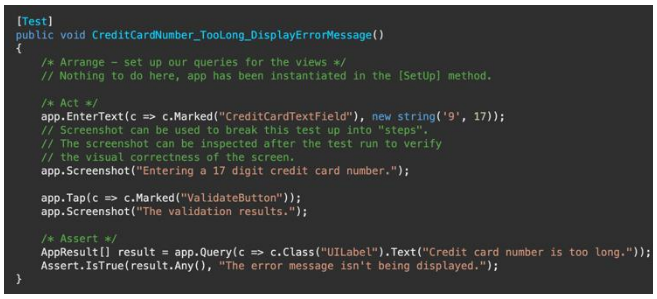


```
void main() {
  testWidgets('MyWidget has a title and message', (WidgetTester tester) async {
    await tester.pumpWidget(const MyWidget(title: 'T', message: 'M'));
    final titleFinder = find.text('T');
    final messageFinder = find.text('M');

    // Use the `findsOneWidget` matcher provided by flutter_test to verify
    // that the Text widgets appear exactly once in the widget tree.
    expect(titleFinder, findsOneWidget);
    expect(messageFinder, findsOneWidget);
  });
}
```

Fig. 8.Flutter widget testing example.

Xamarin uses XamarinUI.Test tool which is a testing framework in C# language using NUnit as its layer. This testing is performed separately for both iOS and Android platforms. The test class example is given in fig 9.



```
[Test]
public void CreditCardNumber_TooLong_DisplayErrorMessage()
{
    /* Arrange - set up our queries for the views */
    // Nothing to do here, app has been instantiated in the [SetUp] method.

    /* Act */
    app.EnterText(c => c.Marked("CreditCardTextField"), new string('9', 17));
    // Screenshot can be used to break this test up into "steps".
    // The screenshot can be inspected after the test run to verify
    // the visual correctness of the screen.
    app.Screenshot("Entering a 17 digit credit card number.");

    app.Tap(c => c.Marked("ValidateButton"));
    app.Screenshot("The validation results.");

    /* Assert */
    AppResult[] result = app.Query(c => c.Class("UILabel").Text("Credit card number is too long."));
    Assert.IsTrue(result.Any(), "The error message isn't being displayed.");
}
```

Fig. 9.Form testing using XamarinUI.Test tool.

Table 1: Hybrid App Frameworks Comparison

Features	**Framework**			
	ReactNative	***Ionic***	***Flutter***	***Xamarin***
Language	Javascript	Typescript	Dart	C#
GUI	Native UI	HTML, CSS	Widgets	Native UI
Code Reusability	90%	98%	50-90%	98%
Hot reload	Yes	No	Yes	No
Testing	Jest	Jest	Test (Flutter)	XamarinUI.Test
Performance	Native like	Moderate	Native like	Moderate

V. CONCLUSION

We looked at several factors to compare the different frameworks, with each one having its own set of features and advantages they offer. The comparison has been summarized in table 1. React Native offers a more native-style app component to be made, and therefore allows the app to look more native-like. Ionic provides a large set of components that can be directly used in the applications, and it offers more of a web-like interface. Flutter is completely different and uses its own set of widgets to design the UI which helps it to have better control over the entire interface. While looking at other features such as operating system support, almost all frameworks seem to support Android and iOS which are the market leaders in terms of device availability and support. We analyzed the four frameworks on different fronts which included their popularity, language support, tools and IDEs available, ease of UI design in the applications, code reusability, and testing. Each of them offered one or other benefits.

The implementation of native app features to be used was also compared. We found that UI development can be easy in Ionic and React Native for developers coming from a web development background, while Flutter and Xamarin can be a steep learning curve for them. For a successful cross-platform framework, there is a lot of scope in terms of the user interface, bringing performance as close to native apps, and supporting hardware features provided by the device. Advanced research and findings are necessary to develop a robust and all-use-based hybrid app development framework.

REFERENCES

1. Samkange-Zeeb, F., and M. Blettner. "Emerging aspects of mobile phone use." *Emerging Health Threats Journal* 2.1 (2009): 7082.
2. A. Almisreb, H. . Hadžo Mulalić, N. Mučibabić, and R. Numanović, “A review on mobile operating systems and application development platforms”, *Sustainable Engineering and Innovation*, vol. 1, no. 1, pp. 49-56, Jun. 2019.
3. Priya Toppo, Tripti Dhote,. (2021). PREFERENCE OF MOBILE PLATFORMS: A STUDY OF iOS VS ANDROID. *International Journal of Modern Agriculture*, *10*(2), 1757 - 1764
4. Xanthopoulos, Spyros, and Stelios Xinogalos. "Mobile app development in HTML5." *AIP Conference Proceedings*. Vol. 1648. No. 1. AIP Publishing LLC, 2015.
5. Majchrzak, Tim A. & Biørn-Hansen, Andreas & Grønli, Tor-Morten. (2017). Comprehensive Analysis of Innovative Cross-Platform App Development Frameworks. 10.24251/HICSS.2017.745.
6. Denko, Blaž & Pecnik, Spela & Fister jr, Iztok. (2021). A Comprehensive Comparison of Hybrid Mobile Application Development Frameworks. International Journal of Security and Privacy in Pervasive Computing. 13. 78-90. 10.4018/IJSPPC.2021010105.
7. Kuitunen, Mika. Cross-Platform Mobile Application Development with React Native. BS thesis. 2019.
8. Yucel, Sakir. "Estimating Cost of Enterprise Mobility Strategy." 2017 International Conference on Computational Science and Computational Intelligence (CSCI). IEEE, 2017
9. Enihe, Raphael & Joshua, Jimmy. (2020). HYBRID MOBILE APPLICATION DEVELOPMENT: A BETTER ALTERNATIVE TO NATIVE. 10.11216/gsj.2020.05.39825.
10. Pinto, Carlos Manso, and Carlos Coutinho. "From native to cross-platform hybrid development." *2018 International Conference on Intelligent Systems (IS)*. IEEE, 2018.
11. Waranashiwar, Juilee, and Manda Ukey. "Ionic Framework with Angular for Hybrid App Development." *International Journal of New Technology and Research* 4.5 (2018): 263068.
12. Hu, Hanyang, et al. "Studying the consistency of star ratings and reviews of popular free hybrid Android and iOS apps." *Empirical Software Engineering* 24.1 (2019): 7-32.
13. Payne, Rap. "Developing in Flutter." *Beginning App Development with Flutter*. Apress, Berkeley, CA, 2019. 9-27.

14. Wasilewski, Kamil, and Wojciech Zabierowski. "A Comparison of Java, Flutter and Kotlin/Native Technologies for Sensor Data-Driven Applications." *Sensors* 21.10 (2021): 3324.
15. Vishal, Kumar, and Ajay Shriram Kushwaha. "Mobile Application Development Research Based on Xamarin Platform." *2018 4th International Conference on Computing Sciences (ICCS)*. IEEE, 2018.
16. Zelenchuk, Denys. "Getting Started with Espresso for Android." *Android Espresso Revealed*. Apress, Berkeley, CA, 2019. 1-48.

AUTHORS PROFILE

Mohit Singh, is an undergraduate student who is pursuing Computer Science & Engineering at R.V. College of Engineering, Bengaluru. His area of interest lies in Computer Vision and App Development. He has 1 publication in his name in the Computer Vision domain.

Dr. Shobha G, is a professor in R. V. College of Engineering, Bengaluru. Her area of interest lies in data mining, image processing, and networking. She has a teaching experience of 25 years and research experience of 14 years. She has 145 publications in international journals and conferences. She has guided 30 UG projects, 40 PG projects and 7 Ph.D. students. She also has 4 patents and has been a reviewer of several books.