Tren dan Peluang Cross-Platform Mobile App untuk Developer Pemula

M Ilhami

STMIK Mikroskil, Medan, Indonesia

E-mail: mirza.ilhami@mikroskil.ac.id

Abstrak. Perkembangan aplikasi mobile berbasis *cross-platform* telah berkembang dengan pesat sejak lima tahun belakangan ini. Dalam sudut pandang *developer*, banyaknya teknologi yang harus dipelajari membuat mereka bingung untuk memilih bahasa pemrograman apa yang harus dikuasai karena akan berkorelasi dengan waktu yang dibutuhkan dan adopsi teknologi oleh dunia industri. Dari sudut pandang industri sendiri, tantangan yang dihadapi adalah menentukan *framework* atau bahasa pemrograman dan *tool* yang tepat untuk aplikasi mereka. Ini juga berkorelasi dengan biaya dan waktu pengembangan, serta menemukan talenta terbaik yang menguasai teknologi tersebut. Tujuan dari *paper* ini adalah untuk memberikan *insight*, tren dan perspektif kepada *programmer* pemula yang ingin memulai karir di industri *cross-platform mobile app* terkait teknologi yang digunakan dengan melihat hasil survei yang dilakukan oleh Stackoverflow, StateOfJS dan Ionic Framework selama 5 tahun belakangan ini. Sehingga ini akan membantu *programmer* pemula dan industri dalam memilih teknologi dan *framework* yang tepat. Penulis menemukan saat ini JavaScript telah menguasai dari sisi *frontend*, *backend* dan *test tool*. Terkait *cross-platform framework*, kami menemukan Ionic Framework, React Native adalah yang paling banyak digunakan saat ini.

Kata kunci: cross-platform; javascript; ionic framework; react; mobile app

Abstract. The development of cross-platform based mobile applications has grown rapidly in the last five years. From a developer's point of view, the many technologies that must be studied make them confused about which programming language to master because it will correlate with the time required and technology adoption by the industrial world. From an industry point of view, the challenge faced is determining the right framework or programming language and tools for their application. It also correlates with development time and costs, as well as finding the best talent with the technology. The purpose of this paper is to provide insight, trends and perspectives to novice programmers who want to start a career in the cross-platform mobile app industry related to the technology used by looking at the results of surveys conducted by Stackoverflow, StateOfJS and Ionic Framework over the past 5 years. So this will help beginner and industrial programmers in choosing the right technology and framework. The author found that currently JavaScript has mastered the frontend, backend and test tools. Regarding cross-platform frameworks, we find the Ionic Framework, React Native to be the most widely used at the moment.

Keywords: cross-platform; javascript; ionic framework; react; mobile app

1. Pendahuluan

Saat ini ada jutaan aplikasi tersedia di 3 marketplace aplikasi utama: App Store Apple, Play Store Google dan Microsoft Store [1][13]. Ratusan juta *download* terjadi setiap tahunnya [13]. Dengan *revenue* ratusan miliar dolar dihasilkan dari aplikasi tersebut hingga tahun 2020 dan akan terus bertumbuh [15]. Sejalan dengan itu, juga memicu berkembangnya teknologi dan bahasa pemrograman dengan sangat cepat. Dari sisi *programmer*, ini menjadi tantangan karena mereka dituntut untuk bisa mengikuti perkembangan teknologi tersebut. Begitu juga dari segi industri, memilih teknologi akan berkorelasi dengan waktu dan biaya yang dihabiskan selama pengembangan [2]. Secara tradisional, aplikasi *mobile* dikembangkan dengan bahasa pemrograman *native* untuk platform spesifik tertentu seperti Java/Kotlin untuk Android, Swift/Objective-C untuk iOS dan C# untuk Windows Phone. Mengelola beberapa *codebase* untuk platform spesifik tentu akan menyita waktu dan biaya [2].

Munculnya *framework cross-platform* memungkinkan *single codebase* dapat digunakan dan *dideploy* ke beberapa platform sekaligus seperti aplikasi mobile (Android, iOS, Windows), aplikasi web atau PWA (Progressive Web Apps) dan aplikasi desktop (Electron) [3][4]. Ketika berbicara *framework cross-platform*, maka ini tidak merujuk kepada bahasa pemrograman spesifik, *tool* atau *framework* tertentu. Ionic Framework, Titanium, MoSync, Cordova, Phonegap adalah merupakan beberapa yang sering disebut dalam literatur [3]. Beberapa *framework* tersebut dapat dikategorikan dalam pendekatan pengembangan berdasarkan parameter-parameter termasuk antarmuka pengguna *(user interface)*, akses ke fungsionalitas perangkat melalui API dan *plugin* serta teknik kompilasi kode program [5]. Kajian studi lainnya yang juga fokus terhadap *cross-platform* menemukan bahwa PhoneGap, Ionic Framework dan React Native adalah yang paling banyak digunakan saat ini sesuai dari survei mandiri yang mereka lakukan [6].

Untuk lebih memahami bagaimana *insight*, tren dan perspektif dari penggunaan *framework*, bahasa pemrograman dan *tools* dibidang *cross-platform mobile apps* selama 5 tahun belakangan ini, penulis menggunakan data survei yang diterbitkan oleh Stackoverflow, StateOfJS dan Ionic Framework dari tahun 2016 hingga tahun 2020. Survei tersebut dilakukan terhadap para praktisi dan developer di seluruh dunia untuk mengetahui teknologi apa yang mereka gunakan, *framework*, bahasa pemrograman hingga gaji, jenis kelamin, pengalaman bekerja, usia dan lain sebagainya. Tujuan dari *paper* ini adalah untuk memberikan pandangan, tren dan perspektif kepada *programmer* pemula tentang teknologi *cross-platform* yang paling banyak digunakan oleh *developer* selama 5 tahun belakangan ini. Dengan mengetahui perkembangan teknologi tersebut, diharapkan programmer pemula lebih dapat menentukan teknologi apa yang harus mereka kuasai dan fokus

2. Kajian Pustaka

2.1. Perbandingan Framework Cross-Platform Mobile

Framework *cross-platform apps* atau juga disebut *hybrid apps* merupakan gabungan dari solusi *native* dan web [14][17]. Disebut *hybrid* karena dengan *framework* ini memungkinkan pengembangan aplikasi ditulis dengan satu bahasa *(one codebase)* namun dapat di-*deploy* ke berbagai *platform* sekaligus, diantaranya Android, iOS, Windows, PWA *(Progressive Web Apps)*, *desktop apps* dan *web apps* [7][17]. Dimana antarmuka *front-endnya* sendiri ditulis dengan bahasa teknologi web, seperti HTML, CSS dan JavaScript [14]. Bagi *programmer* pemula, memilih bahasa pemrograman apa yang harus dikuasai untuk membangun sebuah aplikasi *mobile* terkadang menjadi tantangan sendiri. Ini dikarenakan banyak bahasa pemrograman yang dapat digunakan untuk membangun aplikasi *mobile*, mulai dari bahasa *native* Swift/Objective-C untuk iOS, Java/Kotlin untuk Android dan C# untuk Windows Phone. Selain native, juga tersedia framework *hybrid* seperti Ionic dari Drifty, Xamarin dari Microsoft, Flutter dari Google, NativeScript dari Telerik dan React Native dan Facebook. Framework *hybrid* ini sendiri didukung oleh beberapa *web framework*, kebanyakan dibangun dengan JavaScript seperti Angular, React, Vue, Svelte, Vanilla dan C#. Dalam *paper* ini penulis akan fokus terhadap tren dan tantangan dari teknologi *cross-platform mobile app* dibanding dengan *native mobile app*.

2.2. Native vs Cross-Platform

Bagi perusahaan, penting untuk diketahui bahwa keputusan dalam memilih *hybrid* atau *native* harus berdasarkan pada tujuan dari perusahaan terhadap projek tersebut dan komposisi dari tim development yang tersedia. Sedangkan bagi *mobile app developer*, memilih menguasai teknologi *hybrid* atau *native* akan sangat berpengaruh pada karir dan peluang kedepan. Memilih pendekatan *native*, berarti aplikasi dikembangkan pada setiap *platform* tersendiri, terkadang juga secara spesifik untuk *tablet* atau *smartphone*. Ini berarti setiap platform memiliki *codebase* masing-masing untuk masing-masing *platform*, yang tentunya membutuhkan *resource programmer* yang secara spesifik pula.

Gambar 1. Pendekatan *native app*

Dari Gambar 1 dapat dilihat bahwa setiap tim bertanggung jawab dalam mengembangkan masing-masing platform, tim iOS (Swift/Objective-C) mengembangkan aplikasi untuk iOS sedangkan tim Android (Java/Kotlin) mengembangkan aplikasi Android, begitu juga website dan desktop app. Tentu ini akan sangat berbanding lurus dengan jumlah tenaga ahli, biaya dan waktu yang semakin tinggi [2]. Sedangkan *cross-platfrom framework* memungkinkan jumlah tim yang lebih sedikit untuk mengembangkan aplikasi diberbagai *platform*.

Gambar 2. Pendekatan *cross-platform*

Gambar 2 menunjukkan bagaimana tim web dapat mengembangkan aplikasi untuk berbagai platform iOS, Android, Website dan Desktop. Ini memudahkan mereka dalam pengelolaan karena hanya

menggunakan *single codebase*. Dari segi waktu biaya dan waktu tentu akan jauh lebih hemat dibandingkan dengan *native apps.*

Gambar 3. Perbandingan web dan native iOS & Android

Berdasarkan survei dari Stackoverflow tahun 2019, pertumbuhan web developer 10 kali lebih banyak dibandingkan *mobile native developer*. Juga, kurang dari 7% dari semua developer yang memilih Swift, Kotlin dan Objective-C sebagai bahasa yang familiar. Kontras dengan *web developer* yang mencapai 70% [8].

2.3. Cross-Platform Native: Native UI, Shared Code

Tidak semua teknologi *cross-platform* sama. Aplikasi seperti Instagram menggunakan React Native yang berjalan di *platform* Android dan iOS. Begitu juga dengan aplikasi seperti Google yang menggunakan Flutter dan berjalan juga di Android dan iOS. Namun, ada perbedaan jika dibandingkan dengan Ionic Framework. React Native, Xamarin, NativeScript memungkinkan untuk mengembangkan antarmuka aplikasi menggunakan satu bahasa yang sama yang kemudian diterjemahkan ke komponen berbeda saat *runtime process.* Misal untuk menampilkan komponen text, maka yang terjadi saat *runtime process* adalah: [TextView] untuk Android dan [UIView] untuk iOS, ini berarti tidak berbagi komponen namun berbagi kode program di platform yang berbeda. Jumlah kode yang dapat berbagi dengan platform berbeda sangat bergabung pada kustomisasi aplikasi. Jika tetap pada komponen UI fundamental seperti View, Text, Image dan Touchable maka kode program dapat berjalan disetiap platform, artinya 100% kode program dapat berbagi. Namun, jika terdapat banyak kustomisasi native, maka akan membutuhkan tidak codebase: dua diantaranya adalah untuk mengelola antarmuka Android dan iOS dan sisanya adalah berbagi kode program untuk *controller*. Tim mobile app dari Airbnb menjelaskan bagaimana aplikasi mereka dalam menggunakan React Native: “Even though code in React Native features was almost entirely shared across platforms, only a small percentage of our app was React Native” [9]. Berikut ini adalah bagaimana perbandingan dari native, *framework cross-platform native* dan *framework cross-platform web* [10][16].

Tabel 1. Perbandingan native, cross-platform native, cross-platform web

	Native	Cross-Platform Native	Cross-Platform Web
Contoh	iOS and Android SDKs	React Native, Xamarin, Flutter, NativeScript	Ionic Framework
Bahasa	Obj-C, Swift, Java	JavaScript + Custom UI Language / Interpreter	HTML + CSS + JavaScript

	Native	Cross-Platform Native	Cross-Platform Web
Code Reuse	Terpisah per platform	Berbagi bisnis logik dengan *codebase* antarmuka terpisah	Satu *codebase* yang sama
Target Platforms	iOS & Android Native Mobile Apps	iOS & Android Native Mobile Apps	iOS, Android, Electron, Mobile & Desktop Browsers sebagai PWA *(Progressive Web App)*, dan dimanapun web berjalan
Akses API	Native API terpisah untuk setiap platform	Satu *codebase* yang sama melalui plugin	Satu *codebase* yang sama melalui plugin
Element UI	Native UI untuk setiap platform	Spesifik native UI untuk iOS dan Android. Kustomisasi elemen UI membutuhkan *codebase* terpisah untuk setiap platform	Berbagi element web UI untuk setiap platform
Performansi	Performansi native, cepat	Relatif, berbeda disetiap platform	Relatif, berbeda disetiap platform

Kemiripan antara *cross-platform native* dan *cross-platform web* adalah mereka sama-sama menggunakan bahasa yang sama *(single codebase)* untuk membangun aplikasi yang dapat berjalan diplatform berbeda. Misalnya React Native menggunakan framework React JavaScript, yang dapat me-*render* antarmuka native pada proses *runtime*, memungkinkan untuk membangun aplikasi Android dan iOS. Sedangkan Ionic, menggunakan HTML + CSS + JavaScript untuk membangun aplikasi Android, iOS, PWA, Desktop.

2.4. Cross-Platform Web: Web UI, Shared Components

Dibandingkan menggunakan JavaScript sebagai jembatan untuk membangun antarmuka *native*, komponen antarmuka sebenarnya berjalan disemua platform. Misal di aplikasi *mobile*, komponen antarmuka ini berjalan di layer *webview*. Untuk PWA sendiri, komponen berjalan di browser, sedangkan untuk aplikasi *native* desktop, komponen berjalan di *desktop container* seperti Electron. Dengan pendekatan *cross-platform web*, komponen antarmuka dapat dibangun dengan HTML + CSS + JavaScript dengan tentu saja ketersediaan fungsionalitas native yang dapat diakses melalui API atau plugin, tergantung dari *platform*. Artinya hanya fitur-fitur perangkat *native* saja misalnya Kamera yang bergantung terhadap platform, selebihnya yaitu komponen dapat berbagi untuk semua platform. Karenanya pendekatan ini memungkinkan untuk menggunakan single *codebase* untuk berbagai platform berbeda, memungkinkan tim *developer* untuk mengelola *codebase* yang sama untuk setiap perubahan dan penambahan fitur dari aplikasi yang sedang dikembangkan. Bahkan untuk komponen kustomisasi antarmuka juga dapat dibagi kesemua platform yang berbeda. Dari segi *debugging*, juga memudahkan tim developer karena mereka dapat menggunakan *tools debugging* berbasis web misalnya Chrome Developer tools untuk dapat melakukan *debug* di semua platform.

3. Hasil dan Pembahasan

3.1. Tren Cross-Platform Mobile Apps Framework

Survei kuesioner yang digunakan dalam paper ini bersumber dari survei Stackoverflow, StateOfJS dan Ionic Framework. Ketiganya diambil dari survei tahun 2017, 2018, 2019, 2020. Responden terdiri dari praktisi *(software developer)* dari berbagai industri. Mulai dari *full-stack web developer*, *front-end developer, mobile developer, backend developer, devops*, *data scientist* dan lainnya. Dengan rentang usia mulai dari dibawah 11 tahun hingga diatas 65 tahun.

3.1.1. Survei Stackoverflow – 2016 hingga 2020

Ini merupakan survei tahunan yang dilakukan Stackoverflow kepada praktisi software diseluruh dunia. Tujuan dari survei ini adalah untuk memberikan insight kepada komunitas tentang tren tools, bahasa, framework, teknologi, perilaku programmer *(developer behaviour)*, gaji dan lainnya. Termasuk didalamnya juga teknologi dan framework yang saat ini banyak digunakan dalam cross-platform mobile app. Maka dari itu survei dari Stackoverflow menjadi salah satu rujukan penting dari paper ini. Untuk melihat bagaimana tren dari teknologi yang berkaitan dengan cross-platform mobile app maka digunakan hasil survei Stackoverflow dari tahun 2016 sampai tahun 2020 dengan jumlah responden sebagai berikut.

Tabel 2. Responden Stackoverflow

Tahun	Jumlah responden	Waktu pelaksaan
2016	49397	-
2017	64227	12 Januari – 6 Februari 2017
2018	101592	8 – 28 Januari 2018
2019	88883	23 Januari – 14 Februari 2019
2020	65000	5 – 28 Februari 2020

Jumlah responden dari Tabel 2 cukup bervariasi setiap tahunnya. Adapun 2 teknologi yang paling banyak digunakan dari 5 tahun belakangan ini adalah sebagai berikut.

Tabel 3. Dua teknologi paling popular dari survei Stackoverflow

Tahun	Bahasa	Percent
2016	JavaScript	55.4%
	SQL (or SQL Server)	49.1%
2017	JavaScript	61.9%
	SQL (or SQL Server)	50.8%
2018	JavaScript	69.8%
	HTML	68.5%
	CSS	65.1%
2019	JavaScript	67.8%
	HTML/CSS	63.5%
2020	JavaScript	69.7%
	HTML/CSS	62.4%

Tabel 3 di atas menunjukkan konsistensi dari JavaScript selama 5 tahun menjadi *most popular technologies*. Artinya developer selama 5 tahun ini sangat didominasi oleh *web developer* yang menguasai teknologi web terutama JavaScript, karena JavaScript memang berjalan di web desktop, mobile web dan semua platform yang dapat mendukung web. Walaupun ditahun 2016 dan 2017 SQL menjadi no 2, namun sejak 2018 HTML/CSS telah berada dibawah peringkat JavaScript.

Dataset dari Stackoverflow dari tahun 2016 hingga tahun 2020 dapat diunduh dari: https://insights.stackoverflow.com/survey

3.1.2. Survei StateOfJS – 2016 hingga 2020

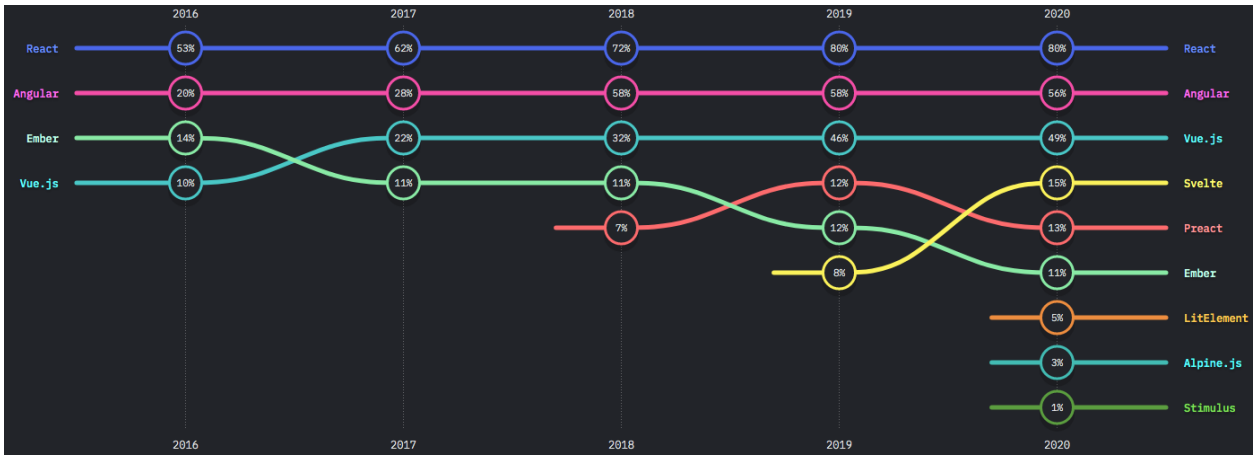
StateOfJS merupakan projek *open-source* yang diinisiasi oleh Sacha Greif, visualisasi dan analisis data dibantu oleh Raphaël Benitte dan Michael Rambeau. Projek ini setiap tahunnya melakukan survei kepada *developer* khusunya JavaScript untuk melihat tren JavaScript dari sisi *front-end frameworks* dan *state management*, hingga *testing library* dan *build tools*. Mereka juga menyajikan data gaji dari para *developer* JavaScript di beberapa negara, *framework* untuk mobile dan desktop serta bahasa pemrograman lainnya. Dikarenakan programmer didunia didominasi oleh *web developer*, maka survei ini cukup penting sebagai

data insight kepada para *programmer* pemula yang ingin berkarir sebagai *software developer* di industri mobile, web dan *desktop apps*.

Tabel 4. Responden StateOfJS

Tahun	Jumlah responsen
2016	9000
2017	28000
2018	20000
2019	21717
2020	23765

Sama seperti yang dilakukan Stackoverflow, Tabel 4 diatas juga menunjukkan jumlah responden yang bervariasi setiap tahunnya. Adapun teknologi JavaScript yang paling banyak digunakan dari 5 tahun belakangan ini adalah sebagai berikut:



Gambar 4. Framework yang paling banyak digunakan dari survei StateOfJS

Gambar 3 diatas menunjukkan React masing memimpin untuk *framework* JavaScript yang paling banyak digunakan sejak tahun 2016, diikuti oleh framework Angular dan Vue.JS yang terus mengalami kenaikan. Di tahun 2020 juga menunjukkan Svelte sedang mengalami kenaikan cukup pesat dari tahun 2019 8% menjadi 15% di 2020. Ini menunjukkan mengapa Ionic Framework bekerja keras untuk mendukung React dan Vue di librari mereka, melihat pengguna React dan Vue yang cukup tinggi.

Dataset dari StateOfJS dari tahun 2016 hingga tahun 2020 dapat diunduh dari: https://www.kaggle.com/sachag/state-of-js

3.1.3. Survei Ionic Framework – 2017, 2018 dan 2020

Survei Ionic Framework dimulai pada tahun 2017, berlanjut setiap tahunnya hingga 2020 kecuali 2019. Survei ini dilakukan kepada komunitas Ionic Framework untuk melihat *insights*, tren dan perspektif dari komunitas tersebut. Developer Ionic didominasi oleh web developer yang menggunakan *javascript-based framework* sebagai bahasa pemrogramannya. Saat ini sendiri, Ionic Framework sudah mendukung Angular, React, Vue dan Vanilla JavaScript untuk mengembangkan aplikasi *cross-platform* yang dapat berjalan di berbagai platform seperti Android, iOS, PWA, Web dan Desktop.

Tabel 5. Responden Ionic Framewok

Tahun	Jumlah responsen
2017	13248
2018	10000
2019	-
2020	1700

Tabel 5 diatas juga menunjukkan jumlah responden yang bervariasi setiap tahunnya, walaupun jumlah respondenya jauh dibawah StateOfJS dan Stackoverlfow. Adapun teknologi yang paling banyak digunakan sejak tahun 2017 di komunitas Ionic Framework adalah sebagai berikut:

Tabel 6. Framework paling banyak digunakan di komunitas Ionic Framework

Tahun	Bahasa	Persentase
2017	Node.js	56.6%
	-	-
2018	Angular	86%
	React	23%
	Vue	17%
2020	Angular	84%
	React	30%
	Vue	20%

Tahun 2017 survei yang dilakukan belum tersedia pertanyaan spesifik terkait dengan *frontend framework*. Untuk *backend framework*, Node.js merupakan yang paling banyak digunakan. Node.js juga merupakan *framework* yang dibangun dengan JavaScript [18]. Ditahun 2018 dan 2020 menunjukkan Angular, React dan Vue mendominasi Svelte dan Vanilla di komunitas Ionic Framework.

Dataset dari Ionic Framework dari tahun 2020 dapat diunduh diakses disini: https://ionicframework.com/survey/2020#introduction

4. Kesimpulan dan Saran

Dengan perkembangan teknologi web yang semakin masif saat ini, meningkatnya produksi *smartphone* dan berbanding lurus juga dengan pemakainya serta tingginya jumlah *web developer* dibandingkan dengan *native app developer*, maka *cross-platform mobile app* dapat menjadi salah satu solusi untuk perusahaan. Sedangkan untuk *programmer* baru yang ingin berkarir di industri ini, *cross-platform framework* dan teknologi web berbasis JavaScript dapat menjadi salah satu alternatif untuk dikuasai dan menjadi ahli dibidang tersebut. Selain itu, perkembangan teknologi *cross-platform mobile app* ini diharapkan juga melahirkan lebih banyak lagi peneliti yang menulis paper tentang teknologi tersebut terutama terkait dengan performansi [11], [12] dan sekuriti.

4.1. Tren & Peluang untuk Mobile App Developer

Melihat data survei yang dilakukan dari 5 tahun belakangan ini dari Stackoverflow, StateOfJS dan Ionic Framework dapat disimpulkan bahwa programmer didunia sangat didominasi oleh web developer. Berdasarkan Stackoverlow, lebih dari 70% mereka ingin disebut sebagai *full-stack developer*. Selain itu juga disisi bahasa pemrograman, JavaScript terus mendominasi setiap tahunnya. Ini diperkuat dengan lahirnya berbagai *framework* berbasis JavaScript seperti Angular/Typescript, ReactJS dan Vue.js. Dominasi *framework based* JavaScript sepertinya akan tetap berlanjut 5 tahun yang akan datang. Melihat tren ini, peluang untuk dapat berkarir di industri mobile dan web sangat terbuka lebar bagi para *developer* pemula dalam memilih bidang keilmuan dan *skillset* yang akan dibangun.

4.2. Adopsi Teknologi untuk Dunia Industri

Solusi dari *cross-platform* native dan *cross-platform web* memberikan kontribusi bagi perusahaan dalam memilih *framework* apa yang cocok dengan kebutuhan. Pemilihan *framework native* atau hybrid akan sangat bergabung dengan tujuan dari aplikasi tersebut dibuat. Untuk menghasilkan aplikasi yang memiliki performansi persis setara native tentu tidak dapat diraih dengan menggunakan *framework cross-platform.* Namun jika perusahaan membutuhkan aplikasi yang dapat berjalan dibanyak platform sekaligus dengan waktu pengembangan yang relatif cepat, biaya dan tenaga ahli yang terbatas, pilihan *cross-platform* dapat menjadi sebuah solusi alternatif tentunya. Dikarenakan, mendapatkan web developer saat ini jauh lebih mudah dibandingkan Android *native developer* dan iOS *native developer*.

Semoga hasil dari ringkasan *paper* ini dapat menambah insight, tren dan perspektif bagi para *developer* pemula yang ingin berkarir pada industri aplikasi *mobile*, memberikan pandangan dan masukan terhadap perusahaan yang ingin mengembangkan aplikasi serta juga memberikan motivasi untuk memperbanyak *paper* dibidang aplikasi mobile kepada peneliti.

5. Ucapan Terima Kasih

Terakhir ucapan terima kasih kepada Stackoverflow, StateOfJS dan Ionic Framework yang telah menyediakan data survei sebagai bahan analisis dari *paper* ini. Juga, terima kasih kepada Konstelasi yang sudah memberikan kesempatan untuk dapat mempublish *paper* ini.

6. Referensi

[1] H. Wang *et al.*, "An explorative study of the mobile app ecosystem from app developers' perspective," *26th Int. World Wide Web Conf. WWW 2017*, pp. 163–172, 2017, doi: 10.1145/3038912.3052712.

[2] P. R.M.de Andrade, A. B.Albuquerque, O. F. Frota, R. v Silveira, and F. A. da Silva, "Cross Platform App : A Comparative Study," *International Journal of Computer Science and Information Technology*, vol. 7, no. 1, pp. 33–40, Feb. 2015, doi: 10.5121/ijcsit.2015.7104.

[3] M. Willocx, J. Vossaert, and V. Naessens, "Comparing performance parameters of mobile app development strategies," May 2016, doi: 10.1145/2897073.2897092.

[4] C. Escoffier and P. Lalanda, "Managing the Heterogeneity and Dynamism in Hybrid Mobile Applications," Jun. 2015, doi: 10.1109/SCC.2015.20.

[5] W. S. El-Kassas, B. A. Abdullah, A. H. Yousef, and A. M. Wahba, "Taxonomy of Cross-Platform Mobile Applications Development Approaches," *Ain Shams Engineering Journal*, vol. 8, no. 2, Jun. 2017, doi: 10.1016/j.asej.2015.08.004.

[6] A. Biørn-Hansen, T. M. Grønli, G. Ghinea, and S. Alouneh, "An Empirical Study of Cross-Platform Mobile Development in Industry," *Wireless Communications and Mobile Computing*, vol. 2019, 2019, doi: 10.1155/2019/5743892.

[7] S. Xanthopoulos and S. Xinogalos, "A comparative analysis of cross-platform development approaches for mobile applications," in *Proceedings of the 6th Balkan Conference in Informatics on - BCI '13*, 2013, p. 213, doi: 10.1145/2490257.2490292.

[8] Stackoverflow, "Stackoverflow 2019 Survey," 2019. https://insights.stackoverflow.com/survey/2019 (accessed Mar. 04, 2021).

[9] Gabriel Peal, "Sunsetting React Native," Jun. 2018. https://medium.com/airbnb-engineering/sunsetting-react-native-1868ba28e30a (accessed Mar. 03, 2021).

[10] Matt Kremer, "Comparing Cross-Platform Frameworks." https://ionicframework.com/resources/articles/ionic-vs-react-native-a-comparison-guide (accessed Mar. 03, 2021).

[11] I. Malavolta, G. Procaccianti, P. Noorland, and P. Vukmirovic, "Assessing the Impact of Service Workers on the Energy Efficiency of Progressive Web Apps," May 2017, doi: 10.1109/MOBILESoft.2017.7.

[12] I. Malavolta, K. Chinnappan, L. Jasmontas, S. Gupta, and K. A. K. Soltany, "Evaluating the impact of caching on the energy consumption and performance of progressive web apps," in *Proceedings - 2020 IEEE/ACM 7th International Conference on Mobile Software Engineering and Systems, MOBILESoft 2020*, Jul. 2020, pp. 109–119, doi: 10.1145/3387905.3388593.

[13] M. A. Harris, R. Brookshire, and A. G. Chin, "Identifying factors influencing consumers' intent to install mobile applications," *Int. J. Inf. Manage.*, vol. 36, no. 3, pp. 441–450, Jun. 2016, doi: 10.1016/j.ijinfomgt.2016.02.004.

[14] W. Jobe, "Native Apps Vs. Mobile Web Apps," *Int. J. Interact. Mob. Technol.*, vol. 7, no. 4, p. 27, Oct. 2013, doi: 10.3991/ijim.v7i4.3226.

[15] S. Baghbaniyazdi and H. Ferdosara, "The Most Successful Business Model of Mobile Applications: A Comparative Analysis of Six Iranian Mobile Games," *J. Softw.*, vol. 12, no. 4, pp. 201–211, 2017, doi: 10.17706/jsw.12.3.201-211.

[16] C. Rieger and T. A. Majchrzak, "Towards the definitive evaluation framework for cross-platform app development approaches," *J. Syst. Softw.*, vol. 153, pp. 175–199, 2019, doi: 10.1016/j.jss.2019.04.001.

[17] I. Malavolta, "Beyond Native Apps: Web Technologies to the Rescue! (Keynote)," *Mobile! 2016 - Proc. 1st Int. Work. Mob. Dev. co-located with SPLASH 2016*, pp. 1–2, 2016, doi: 10.1145/3001854.3001863.

[18] F. Kaimer and P. Brune, "Return of the JS: Towards a node.js-based software architecture for combined CMS/CRM applications," *Procedia Comput. Sci.*, vol. 141, pp. 454–459, 2018, doi: 10.1016/j.procs.2018.10.143.