Numa Head of Engineering

Kosawat Sukchaya
October 2022

Executive Summary

Task#1: Mobile Application

- Considering time and cost of development and building the team, together with the requirements of the app, and the expertise of current team (JavaScript, React), **React Native** could be the best option for Mobile Application Framework.
- New Mobile Application team could be setup with the **mix of new hirings** (Senior / Lead Mobile developer, QA, PM) with **existing Full-Stack / React developers**.
- Before making final decision, the **survey** could be also done within the current teams regarding the **Mobile Application Framework** and **willingness** to learn / join mobile app development team.

Executive Summary

Task #2: Test Coverage

- Firstly, measure and find out what is exactly the testability of the team.
- Bridging knowledge gap by providing team training, encouraging knowledge sharing and establishing well defined test documentations.
- Improving planning, ensuring time slots for improving test coverage related tasks.
- Clear communication about benefits and trade-offs of improving test coverage in order to get buy-in from the teams and stakeholders.

Task #1: Mobile Application

Overview - Requirements

Summary of requirements

- To roll out a mobile application to further boost customer retention & brand awareness.
- The app should cover similar functionality as the existing.
- Thinking long term the app might become more relevant [in terms of usage / revenue generated / guest retention] than our existing webapps.

Overview - Current Teams Setup

- The “product” teams focus on every product that our guests can interact with.
- The “platform” teams focus on abstracting complex business logic and systems (+ internal frontends for customer support & operation teams).
- Besides those teams there are shared resources for QA, DevOps & data engineering.

Overview - Current Teams Setup

5 Teams + Share resources

Product #1	Product #2	Product #3	Platform #1	Platform #2
• PM • Lead Engineer • 2-3 Full stack engineers	• PM • Lead Engineer • 2-3 Full stack engineers	• PM • Lead Engineer • 2-3 Full stack engineers	• PM • Lead Engineer • 2-3 Full stack engineers	• PM • Lead Engineer • 2-3 Full stack engineers

Shared resources: QA, DevOps & data engineering

Mobile Development Options

PWA	Native	Cross-Platform
• Native Script • React • Angular • Vue.js • Ionic	• Swift / Objective C for iOS • Java / Kotlin for Android	• React Native (Facebook, JavaScript, React) • Flutter (Google, Dart) • Xamarin (Microsoft, C#) • Unity (C#)

Criteria for consideration

- **UI / UX** across devices and operating systems
- **Development Cost:** How many new developers we have to hire and how expensive / available are they?
- **Maintenance Cost:** How many code bases to be maintained by the teams?
- **Time to market:** How long does it take to build the team and develop the Mobile Application?
- **Performance:** How fast is it, how is battery consumption?
- **Programming Language:** Learning curve for our current developers, how reusable from our current web application?
- **Maturity:** Community, Support

Comparisons

PWA
• Low Development Cost, Low Maintenance Cost as there could be one code base for all Web app, iOS and Android • No need to build new team • Non native UI/UX • Limited mobile features, need to develop ourselves every needed mobile feature if it cannot be integrated from Native API • More battery consumption • Not available in App stores / Google play stores

Comparisons

Native
• Native UI / UX • Fully support mobile features • Highest performance • Available in App Stores / Google Play Stores • Need to have separated developers and codes base for iOS and Android • Need to hire completely new developers for Swift / Objective-C developers and Java / Kotlin developers • Non reusable code from current web app → longer time to market

Comparisons

Cross-Platform
• Low development and maintenance cost, one code base for iOS and Android • Support relevant mobile features • High performance • Available in App Stores / Google Play Stores • Reusable code from web app in case of **React Native** • Non fully Native UI/UX • It can be rejected from app stores if it does not work smoothly enough • Still need time to develop as a new application

React Native could be an optimum choice

Why React Native?
• React Native is **Cross Platform Mobile Application Framework** • **One code base** for iOS and Android • It is based on **React and JavaScript**. It is possible to reuse some codes / components already implemented in current web app. • React developers can **learn** React Native **quickly**. • **Hiring** React / React Native developers are **more cost effective**, **more available** in the market than e.g Swift, Kotlin, Flutter developers. • React Native is Mature, with **positive community and supports**.

Team building

Mix between new hires and current developers

- A new team for Mobile Application
- New hirings for
  - 1 x Product Manager with experience in Mobile Application
  - 1 x Senior / Lead React Native Developer as the Lead Engineer role
  - 1 x Senior QA with experience in testing Mobile Application
  - (Optional) an addition React Native Developer with at least 2 years of experience
- Transition from current developers
  - Full-Stack React developers who are willing to transfer to the Mobile Application team
  - It is very beneficial to have developers who know in and out of current web application. This will speed up the development and which part of current web application can be reusable.

Team Survey could also be helpful

We may find hidden Champions already in our teams

- We conduct the survey in our teams regarding following topics:
  - Mobile Application Framework preferences
  - Willingness to transition to Mobile Application development team
  - Whether any of them already have experience with Mobile Application
- We could possibly find out that:
  - Our teams might prefer Flutter than React Native
  - Some developers might already have worked with Mobile Application before or they are currently working with it as their side project. They might even willing to transfer and lead the Mobile Application Development team
- We will also conduct an additional session for discussing survey results and brainstorming on Mobile Application Framework topic.

Task #2: Test Coverage

Overview

Summary of problems / issues

- Nearly all teams are **unsatisfied** with the test coverage within their teams
- It seems reasons range from **lack of knowledge** to **lack of time**.

Understand the current situation

Understand the current test coverage

- Measuring the test coverage
  - Against the features and requirements
  - Based on UI or Web elements present
  - Based on User's flow
- How many are untested areas which are critical?
- How many testing levels are currently performed (Unit Testing, Integration Testing, System testing, UAT) ?
- Those information are very helpful for analysing the QA process and Testability in the team

Solutions for Lack of knowledge

Bridging knowledge gap

- Team training
  - Best practices in writing test
  - Testing tools and frameworks
- Encourage knowledge sharing
  - Among cross-functional team members (QA, Developer, DevOps)
  - Among the different teams (Products, Platforms)
- Well defined documentations
  - Shared space as the single source of truth for all test related documentations
  - Clear and well defined test metrics
  - Everyone in the team can contribute to the documents.

Solutions for Lack of time

Improving Test Planning

- “Having not enough time” often caused by poor planning.
- Ensuring to have the tasks related to improving test coverage in the Sprint planning.
- Tasks include, but not limited to, writing unit test, writing test case, refactoring if needed.
- Clear communication inside and outside the team regarding benefits and trade-offs of improving test coverage, for example:
  - Improving test coverage increases Software Quality and thus, improves user satisfaction, reduces downtime, etc.
  - The trade-offs could be having more tasks for improving test coverage results in having less tasks for developing new features

Q&A

Thank you very much!