We were doing
Native Android Development
What else??

Hybrid

MOBILE APPLICATIONS

SYAMKRISHNA BG
NATIONAL INFORMATICS CENTRE NIC

Hybrid mobile applications are built in a similar manner as websites.

Both use a combination of technologies like HTML, CSS, and JavaScript.

However, instead of targeting a mobile browser, hybrid applications target a WebView hosted inside a native container.

This enables them to do things like access hardware capabilities of the mobile device.

A “**webview**” is a browser bundled inside of a mobile application

What are the motivations to go hybrid?
Hybrid mobile applications provide a way for developers to re-use their existing skills in web development.

Native applications have the best performance, highest security, and best user experience.

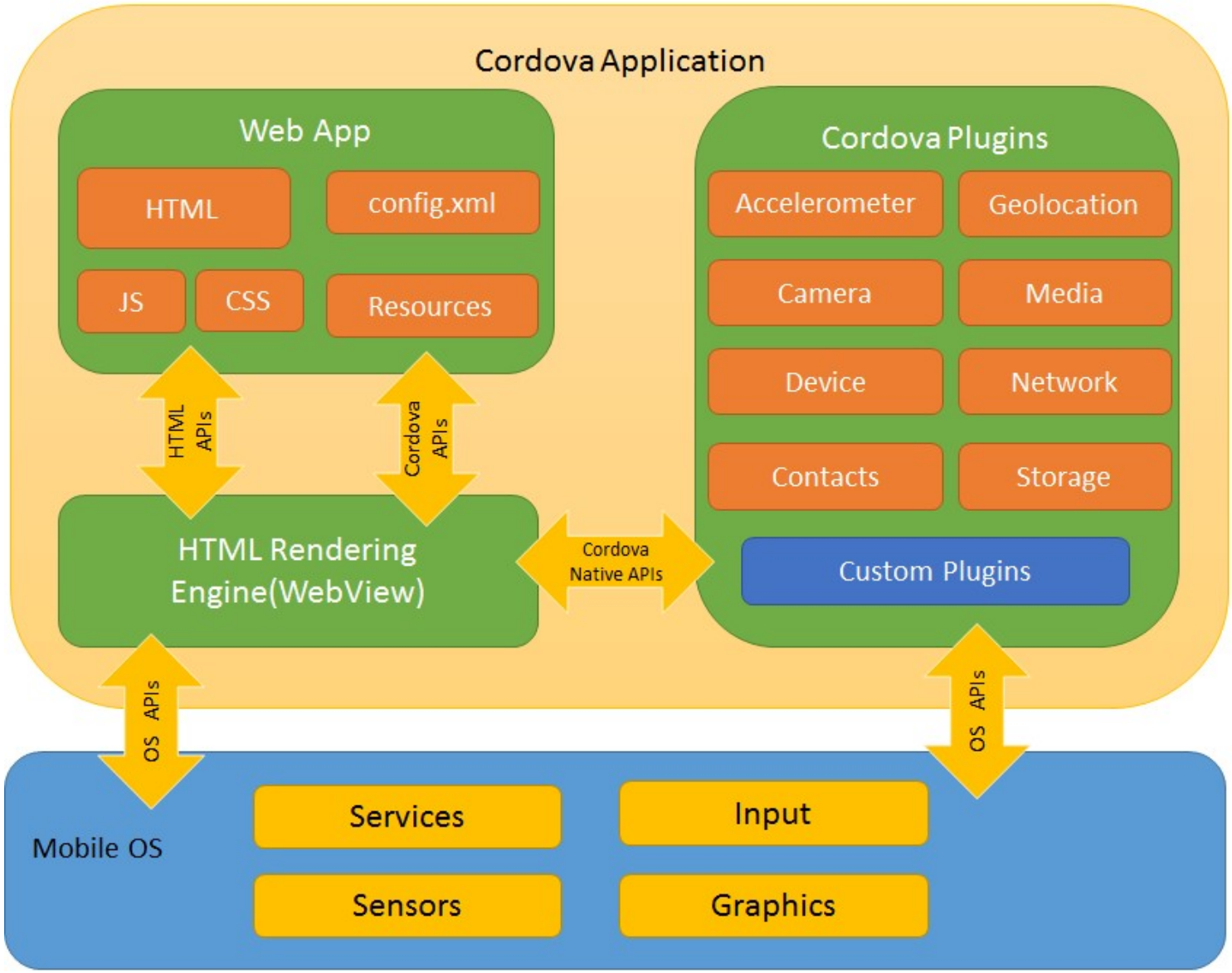
SWOT Analysis by IBM
https://www.ibm.com/developerworks/community/blogs/mobileblog/entry/swot_analysis_hybrid_versus_native_development_in_ibm_worklight?lang=en

Before entering the world of mobile app development you should decide the approach to take: Hybrid or Native?

Hybrid Application Frameworks

- **Xamarin** - Build native apps for multiple platforms on a shared C# codebase. Microsoft-owned .
- **Apache Cordova**
- **Phonegap -** Hybrid application build with HTML, CSS, and JavaScript. Adobe owned. Free PhoneGap is a distribution of Apache Cordova
- **Intel XDK**

etc


Cordova Application
Web App
HTML
config.xml
JS
CSS
Resources
Cordova Plugins
Accelerometer
Geolocation
Camera
Media
Device
Network
Contacts
Storage
Custom Plugins
HTML APIs
Cordova APIs
HTML Rendering Engine(WebView)
Cordova Native APIs
OS APIs
OS APIs
Mobile OS
Services
Input
Sensors
Graphics

pwa

PROGRESSIVE WEB APPLICATIONS

WHAT | WHEN | HOW

NATIONAL INFORMATICS CENTRE

NATIONAL INFORMATICS CENTRE

Evolution of Web

W3Consortium

- HTML
- HTML 2.0
- HTML 3.2
- HTML 4.01
- XHTML
- HTML5

https://www.w3.org/People/Raggett/book4/ch02.html

NATIONAL INFORMATICS CENTRE

- **Evolution of**
  Native Browsers

IE 6,7,,..11

EDGE

CHROME

FIREFOX

OPERA

SAFARI

NATIONAL INFORMATICS CENTRE

- **Evolution of**
  Native Applications

Java

Symbian

Android

iOS

Windows

Blackberry

NATIONAL INFORMATICS CENTRE

- **Evolution of**
  Hybrid Applications

XAMARIN

PHONEGAP

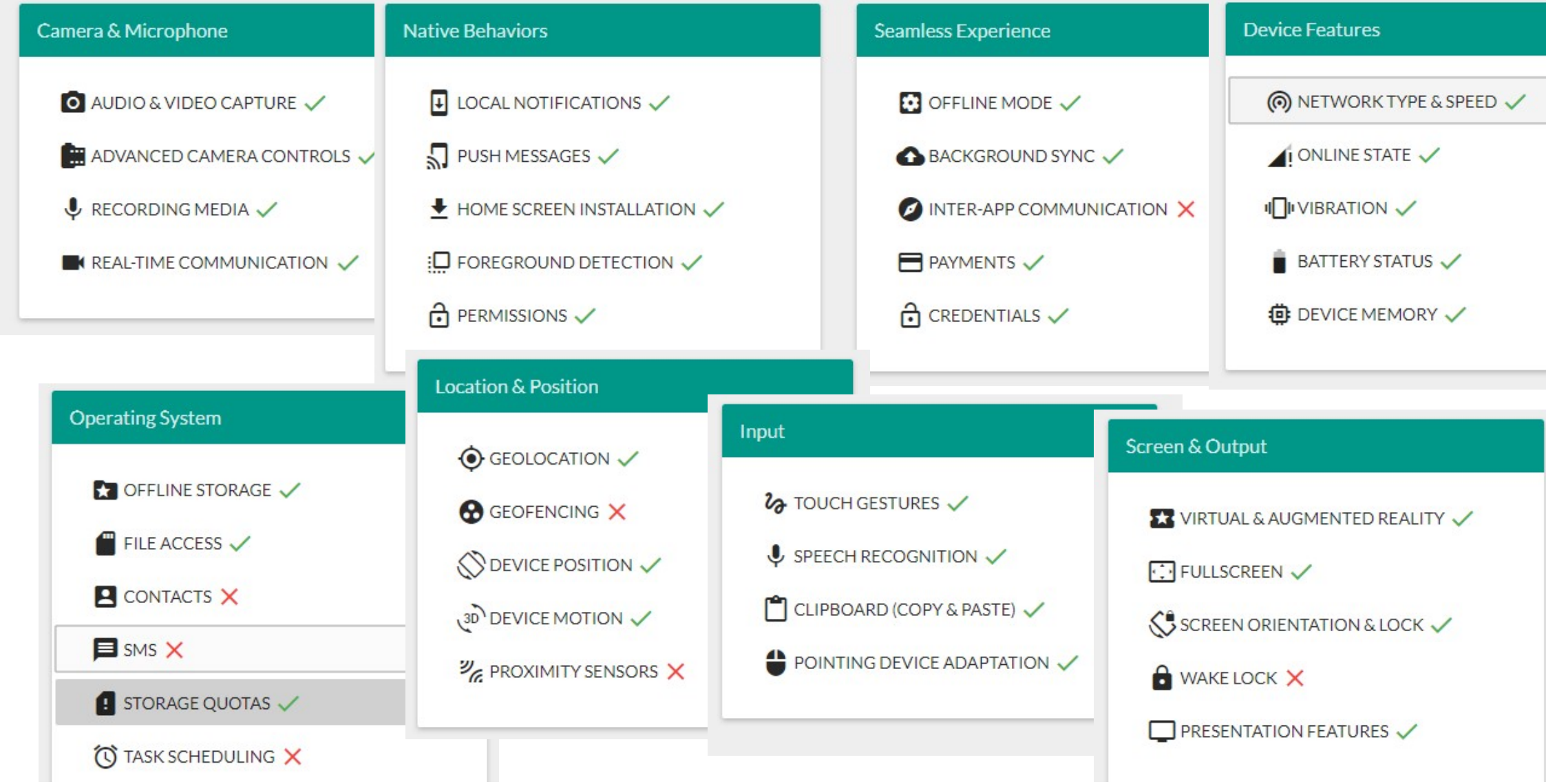
Intel XDK

Ionic

Framework7

Onsen UI

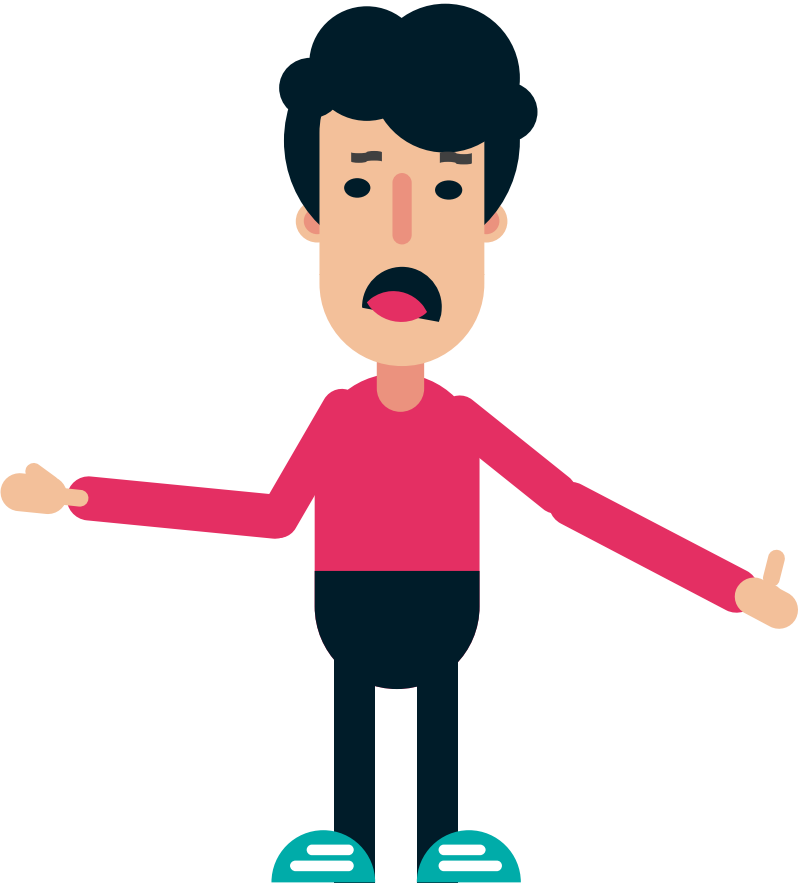
WHAT WEB CAN DO . TODAY



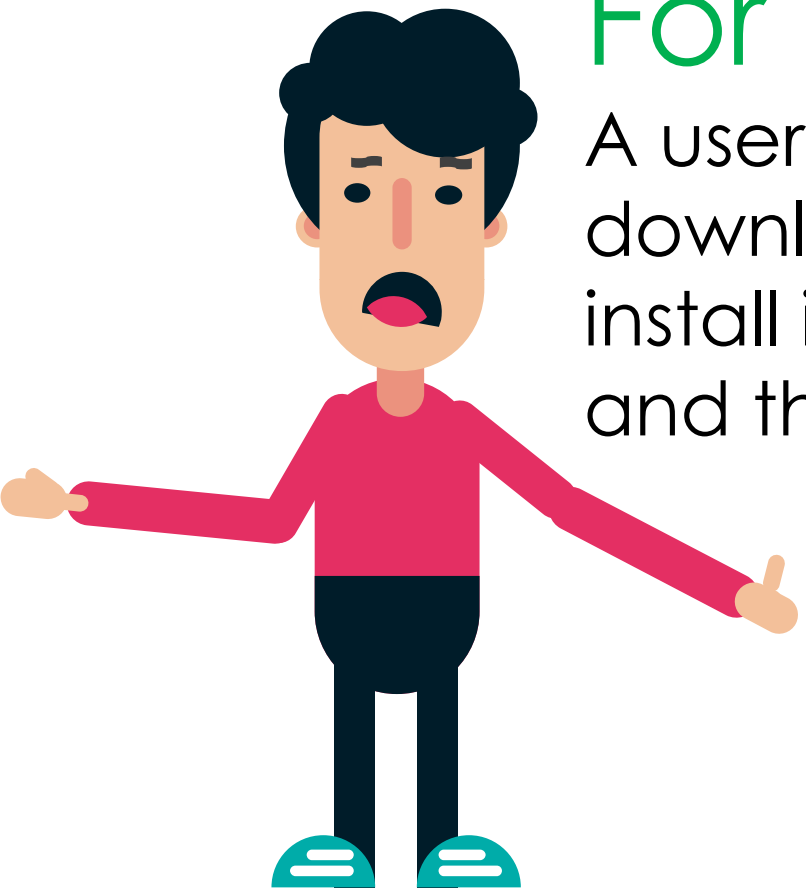
User Experience Matters

Full screen experience
UI blocking

Internet is cheap
Memory has increased



What matters in Conventional Mobile Applications



For end user

A user must first find the app in an app store,
download it,
install it
and then, finally, open it

For developers

a website can be built in less time,
that an API does not need to be maintained with backwards-compatibility
(all users will **run the same version** of your website's code, unlike the version management of native apps)
and that the app will generally be easier to deploy and maintain.

For Management

Maintain separate infrastructure for web and mobile Development

Operating expenses for multiple platforms

What is
Progressive Web App

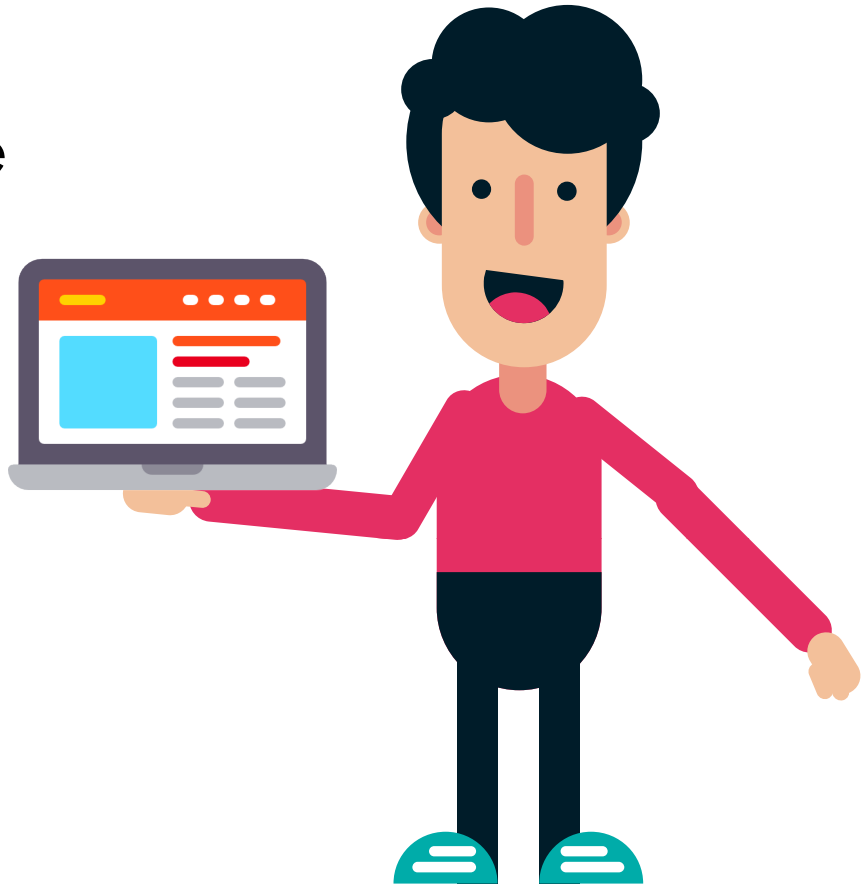
NATIONAL INFORMATICS CENTRE NIC

A progressive web application takes advantage of the latest technologies to **combine the best of web and mobile apps**. Think of it as a website built using web technologies but that acts and feels like an app.

1. Browser Advancements
2. Service Workers
3. Caching
4. Push APIS

Progressive web apps take advantage of the much larger web ecosystem
Ease of deploying and maintaining a website when compared to a native application in the respective app stores.

When a user finds your progressive web app, they will be able to immediately start using it, **eliminating the unnecessary downloading and installation stages**. And when the user returns to the app, they will be prompted to install the app and upgrade to a full-screen experience.



WHEN SHOULD YOU BUILD A PROGRESSIVE WEB APP?

NATIONAL INFORMATICS CENTRE NIC

- When assessing whether your next application should be a progressive web app, a website or a native mobile application, first identify your users and the most important user actions.
- Being “progressive,” a progressive web app works in all browsers, and the experience is enhanced whenever the user’s browser is updated with new and improved features and APIs.
- Thus, there is no compromise in the user experience with a progressive web app compared to a traditional website; however, you may have to decide what functionality to support offline, and you will have to facilitate navigation

- If certain features are required for critical user actions but are not yet available due to a lack of cross-browser support, then a native mobile application might be the better option, guaranteeing the same experience for all users.

Characteristics Of Progressive Web Apps

Characteristics

- **Progressive**. By definition, a progressive web app must work on any device and enhance progressively, taking advantage of any features available on the user's device and browser.
- **Discoverable**. Because a progressive web app is a website, it should be discoverable in search engines. This is a major advantage over native applications, which still lag behind websites in searchability.
- **Linkable**. As another characteristic inherited from websites, a well-designed website should use the URI to indicate the current state of the application. This will enable the web app to retain or reload its state when the user bookmarks or shares the app's URL.

Characteristics

- **Responsive**. A progressive web app's UI must fit the device's form factor and screen size.
- **App-like**. A progressive web app should look like a native app and be built on the application shell model, with minimal page refreshes
- **Connectivity-independent**. It should work in areas of low connectivity or offline (our favorite characteristic).
- **Re-engageable**. Mobile app users are more likely to reuse their apps, and progressive web apps are intended to achieve the same goals through features such as push notifications.

Characteristics

- **Installable**. A progressive web app can be installed on the device's home screen, making it readily available.
- **Fresh**. When new content is published and the user is connected to the Internet, that content should be made available in the app.
- **Safe**. Because a progressive web app has a more intimate user experience and because all network requests can be intercepted through service workers, it is imperative that the app be hosted over HTTPS to prevent man-in-the-middle attacks.

The first characteristic of a progressive web app is that it must work on all devices and must enhance on devices and browsers that allow it.

```
<!DOCTYPE html> <html lang="en">
<head> <meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Airport Arrivals</title>
<link async rel="stylesheet" href="./css/style.css">
<link href="https://fonts.googleapis.com/css?family=Roboto:300,600,300italic,600italic"
rel="stylesheet" type="text/css">
</head>
<body>
    <header>
        <div class="content"> <h3>Arrivals</h3> </div>
    </header>
    <div class="container">
        <div id="main" class="content">
            <ul class="arrivals-list" data-bind="foreach: arrivals">
                <li class="item">
                        <span class="title" data-bind="html: title"></span>
                        <span class="status" data-bind="html: status"></span>
                        <span class="time" data-bind="html: time"></span>
                </li>
            </ul>
        </div>
    </div>
<script src="./js/build/vendor.min.js"></script>
<script src="./js/build/script.min.js"></script>
</body>
</html>
```

For HTML5 Pages

Material design Lite

https://getmdl.io

Javascript Frameworks

React or AngularJS or Knockout

App Like Behaviours

1. Full screen mode support
2. Application launcher icon
3. Installation
4. Offline

Web App Manifest.

1. Full screen mode support
2. Application launcher icon
3. Installation

```
{
  "short_name": "PWA",
  "name": "My PWA",
  "icons": [
    {
      "src": "favicon.ico",
      "sizes": "192x192",
      "type": "image.png",
    }
  ],
  "start_url" : "./index.html",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff",
}
```

MANIFEST.JSON

MANIFEST.JSON

```
{

"short_name": "Arrivals",
"name": "Arrivals at Sky High",
"description": "Progressive web application demonstration",
"icons":
       [
{ "src": "launcher-icon.png", "sizes": "48x48", "type": "image/png" },
{ "src": "launcher-icon-96.png", "sizes": "96x96", "type": "image/png" },
{ "src": "launcher-icon-144.png", "sizes": "144x144", "type": "image/png" },
{ "src": "launcher-icon-192.png", "sizes": "192x192", "type": "image/png" },
{ "src": "launcher-icon-256.png", "sizes": "256x256", "type": "image/png" }
       ],
"start_url": "./?utm_source=web_app_manifest",
"display": "standalone",
"orientation": "fullscreen, standalone, minimal-ui or browser.",
"theme_color": "#29BDBB",
"background_color": "#29BDBB"

}
```

```
<link rel="manifest"  href="./manifest.json">
```

Service Workers

One of the more exciting aspects of progressive web apps is that they can work offline.

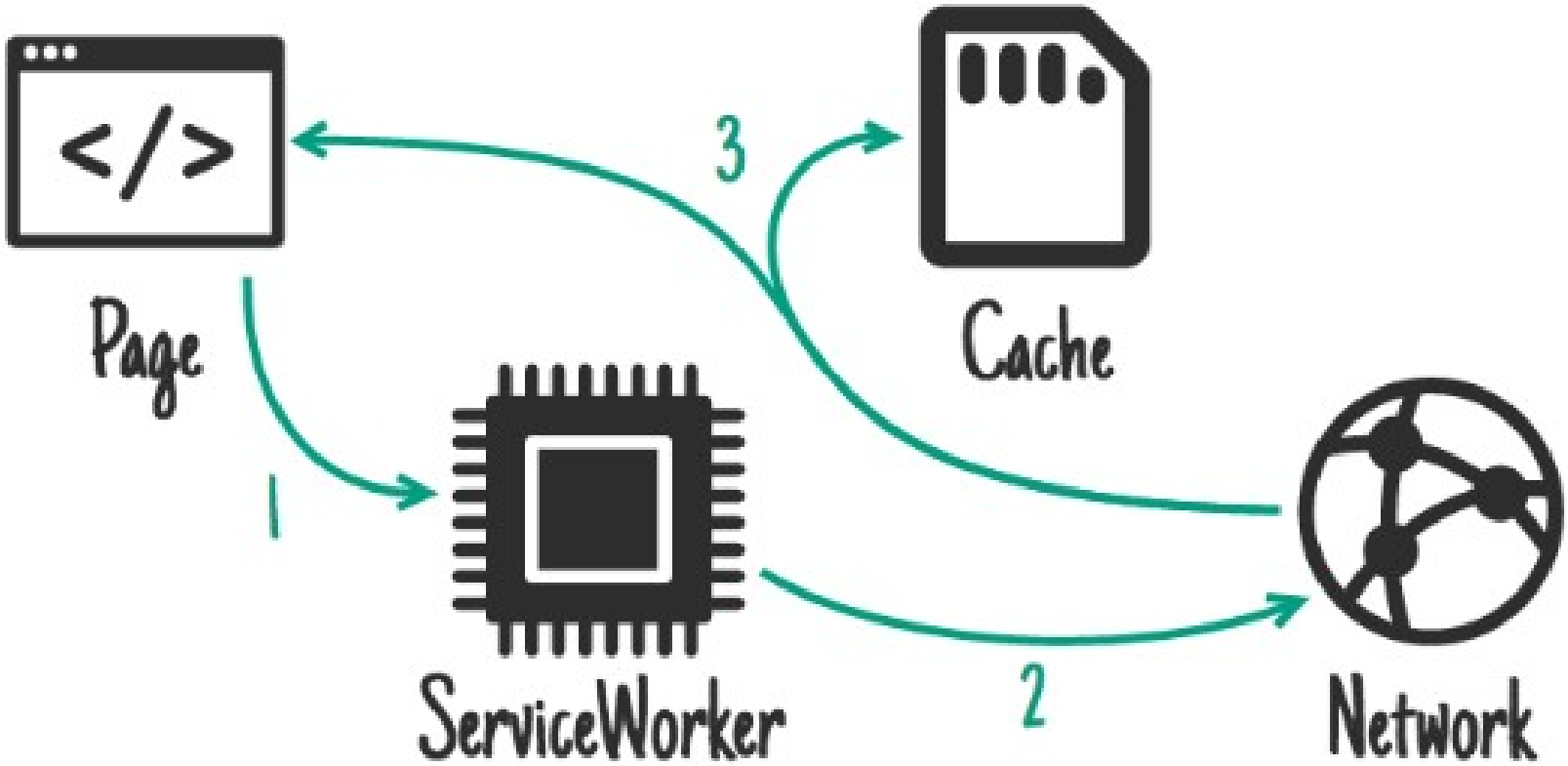
Using service workers, it is possible to show data that was retrieved in previous sessions of the app (using IndexedDB)

or, alternatively, to show the application shell and inform the user that they are not connected to the Internet

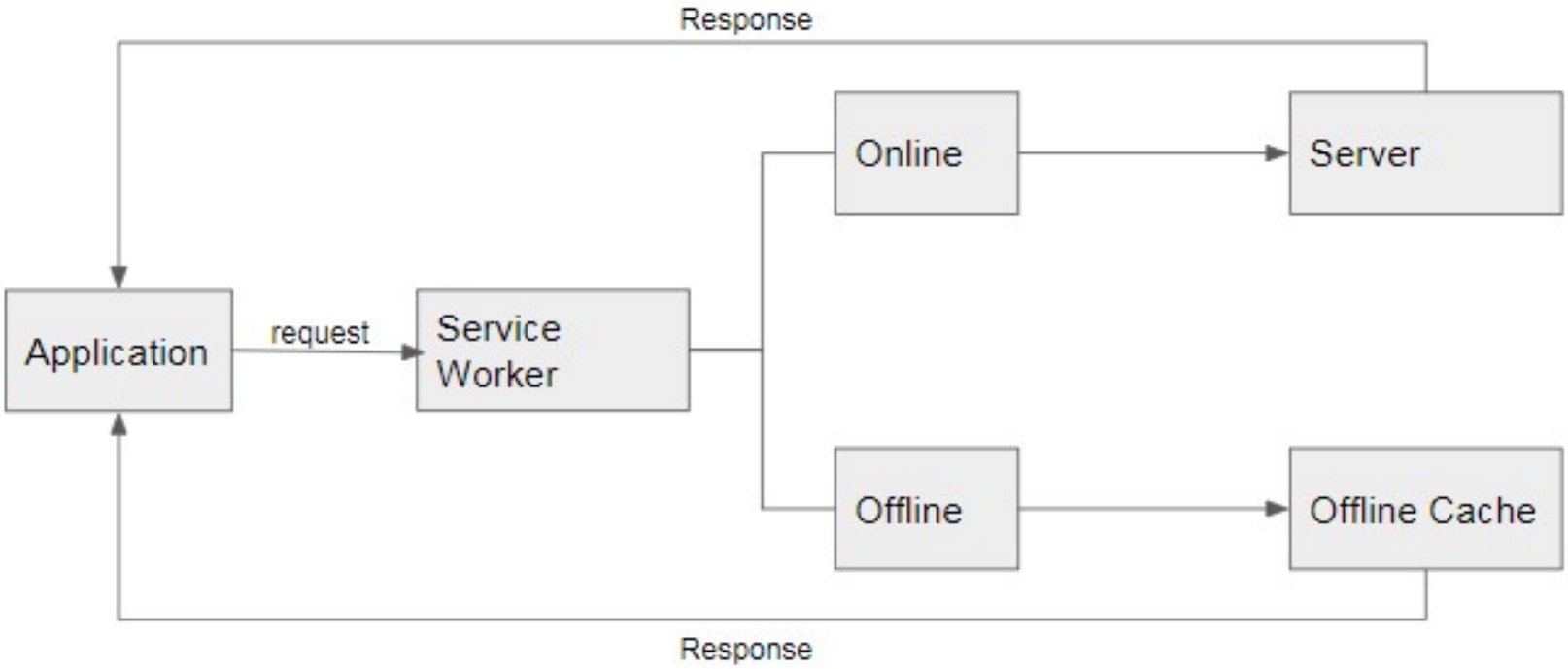
Once the user reconnects, we can then retrieve the latest data from the server.

Service workers are a type of web worker, an object that executes a script separately from the main browser thread

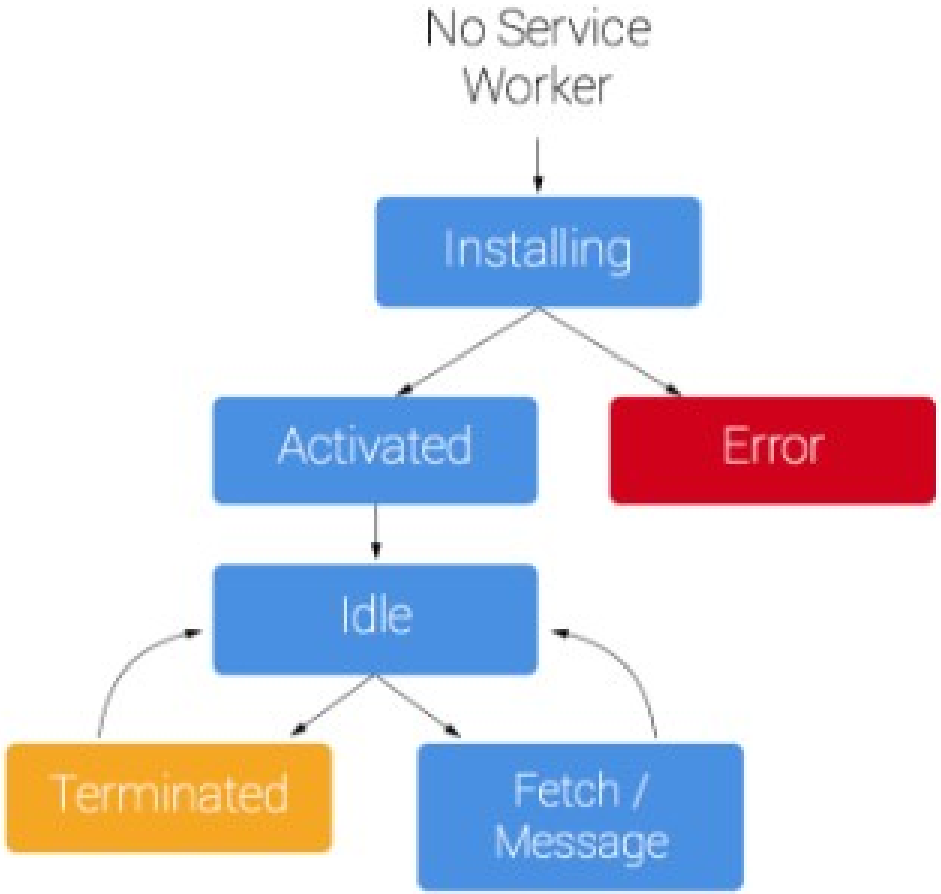
Service Worker Operation



Service Worker Operation



Service Worker Lifecycle

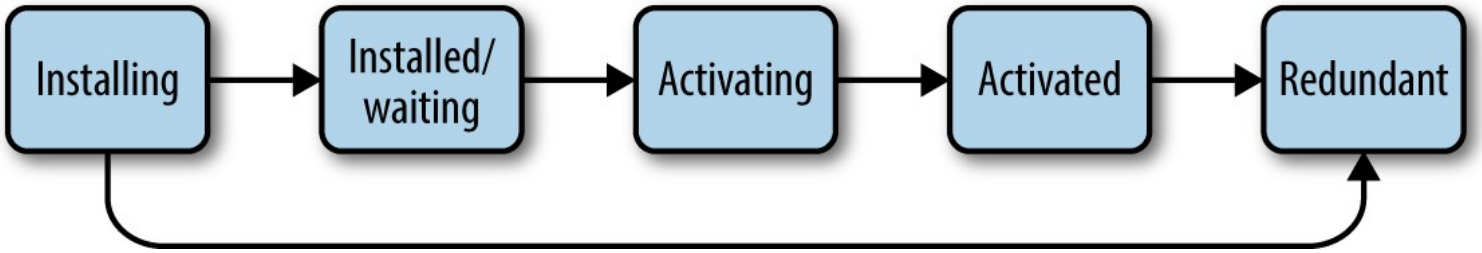


main.js

```
if (!('serviceWorker' in navigator)) {
  console.log('Service Worker not supported');
  return;
}
navigator.serviceWorker.register('/service-worker.js')
.then(function(registration) {
  console.log('SW registered! Scope is:', registration.scope);
}); // .catch a registration error
```

service-worker.js

```
self.addEventListener('install', function(event)
{
  // Do stuff during install
});
```




```
self.addEventListener('activate', function(event)
{
  // Do stuff during activate
});
```

```
self.addEventListener('fetch',
function(event) {
  event.respondWith(
    caches.match(event.request)
  );
});
```

Application Shell

The application shell is the minimum HTML, CSS and JavaScript required to power a user interface.

Push Notifications

The Push API is supported in Chrome, Opera and Samsung's browser and is under development in Firefox and Microsoft Edge.

Local Storage Limits

Browser	Limit
Chrome	<6% of free space
Firefox	<10% of free space
Safari	<50MB
IE10	<250MB
Edge	Dependent on volume size

Browser Support for API

API	Chrome	IE	FIREFOX	SAFARI	OPERA
Web Workers	4.0	10.0	3.5	4.0	11.
SSE	6.0	Not supported	6.0	5.0	11.5

Lighthouse

demo

PROGRESSIVE WEB APPLICATION

NATIONAL INFORMATICS CENTRE

demo

https://
getpwa.github.io/
informatics/

source

https://github.com/
getpwa/informatics

NATIONAL INFORMATICS CENTRE

Thank you