What is pwa?

January 10, 2024.

Contents

1 Introduction **3**
1.1 Purpose . . . 3
1.2 Ámbito del sistema . . . 3
1.3 Acronyms . . . 3
1.4 References . . . 3
1.5 Overview of the Document . . . 4

2 General Desciptions / what is PWA? **4**
2.1 Perspective products . . . 4
2.2 Product functionality . . . 4
2.3 User characteristics . . . 5
2.4 Restrictins . . . 5
2.5 Assumptions and dependencies . . . 5
2.6 Future requirements . . . 5
2.7 Concepts from pwa . . . 6

3 Specific Requirements for PWAs **7**
3.1 External Interfaces . . . 7
3.2 Functions . . . 7
3.3 Performance Requirements . . . 7
3.4 Design Constraints . . . 7

4 SOA (Service-Oriented Architecture) **7**
4.1 advantages . . . 7
4.2 disadvantages . . . 8

5 Native app and Cross-platform **8**
5.1 Native app . . . 8
5.2 Cross-platform . . . 9

6 Appendices **10**

1 Introduction

In this document we will talk about what is a pwa, what are the concepts and information for SOA (Service-Oriented Architecture) and the characteristics from native apps and cross-platform.

1.1 Purpose

The purpose of this document is to provide a detailed description of the pwa and its characteristics and the concepts that are used to create a pwa. then will explain the SOA (Service-Oriented Architecture) and the native apps and cross-platform.

1.2 Ámbito del sistema

En esta subseccion:

- Se podrá dar un nombre al futuro sistema (p.ej. MiSistema)
- Se explicará lo que el sistema hará y lo que no hará.
- Se describirán los beneficios, objetivos y metas que se espera alcanzar con el futuro sistema.
- Se referenciarán todos aquellos documentos de nivel superior (p.e en Industria de sistemas, que incluyen hardware y software, deberían mantenerse la consistencia con el documento de especificaciones de requisitos globales del sistema, si existe)

1.3 Acronyms

In this document, the following acronyms are utilized:

- HTML: HyperText Markup Language
- CSS: Cascading Style Sheets
- Js: JavaScript
- PWA: Progressive Web App
- SOA: Service-Oriented Architecture
- AWS: Amazon Web Services
- API: Application Programming interface

1.4 References

References

[1] Gillis, A. S. (2022, December 8) native app. Software Quality.January 9, 2024, from `https://www.techtarget.com/searchsoftwarequality/definition/native-application-native-app`

[2] Fernández, C. (2022, March 18). Cross-platform app development. Characteristics. ABAMobile. January 9, 2024, from https://abamobile.com/web/cross-platform-app-development-characteristics/

[3] aws (n.d.). what is SOA? - SOA architecture explained. Amazon Web Services, Inc. January 9, 2024, from https://aws.amazon.com/what-is/service-oriented-architecture/

[4] vuestorefront (n.d.). what is PWA? Progressive Web Apps Explained | Vue Storefront. Vue Storefront. January 9, 2024, from https://vuestorefront.io/blog/pwa

1.5 Overview of the Document

This document explores various aspects related to Progressive Web Apps (PWAs), Service-Oriented Architecture (SOA), as well as native apps and cross-platform development. The discussion will delve into the definition and characteristics of PWAs, essential concepts and information pertaining to SOA, and the distinctions between native applications and cross-platform solutions. By examining these topics, readers will gain insights into the fundamental principles and considerations associated with modern application development across different platforms.

2 General Desciptions / what is PWA?

PWA is an app built using web technologies and provides a UX (User Experience). It's like a website can run on multiple platform or devices for a single code, it use Html, css and js.

2.1 Perspective products

Offer a cutting-edge approach to web development. They blend the best of both web and native applications, providing users with a seamless, app-like experience. PWAs are designed to be responsive, reliable, and fast, allowing users to access content even in low-network conditions. They leverage service workers for offline functionality, enabling users to interact with the app without an internet connection.

2.2 Product functionality

PWAs come packed with powerful functionalities that enhance the user experience. Here's a glimpse:

- Offline Functionality: Thanks to service workers, PWAs can work offline, allowing users to access content and features even when they're not connected to the internet.
- Responsive Design: PWAs are built to adapt to various screen sizes and devices, ensuring a consistent and optimal experience on desktops, tablets, and smartphones.
- App-Like Interactions: PWAs mimic the feel of native apps, offering smooth transitions, gestures, and interactions, creating a more immersive and engaging user interface.

2.3 User characteristics

Users of Progressive Web Applications (PWAs) typically appreciate the following characteristics:

- Accessibility: PWAs are easily accessible through web browsers, eliminating the need for app store downloads. Users can simply visit the website and enjoy the PWA experience.
- Cross-Platform Compatibility: PWAs work on various devices and operating systems, providing a consistent experience across desktops, tablets, and smartphones. This versatility appeals to users with diverse devices.

2.4 Restrictins

While Progressive Web Applications (PWAs) offer a range of benefits, they do have some limitations and restrictions:

- Limited Access to Device Features: PWAs have restricted access to certain device features, such as contacts, SMS, and some hardware functionalities. This limitation is in place to ensure user privacy and security.
- Dependence on Browser Support: The success of PWAs relies on browser support for certain features. While major browsers widely support PWAs, some functionalities may not be uniformly available across all platforms.

2.5 Assumptions and dependencies

In the development and implementation of Progressive Web Applications (PWAs), several assumptions and dependencies are crucial for success:

- Browser Compatibility: PWAs depend on browsers supporting the necessary features and functionalities. Developers must consider browser compatibility when creating PWAs to ensure a consistent user experience.
- Network Conditions: Assumptions about users having consistent or intermittent internet connectivity influence the design of offline capabilities in PWAs. The effectiveness of service workers and caching strategies depends on the reliability of network conditions.

2.6 Future requirements

Predicting the future of technology can be tricky, but there are certain trends and potential requirements that may shape the future of Progressive Web Applications like Enhanced Offline Capabilities: As user expectations grow, future PWAs may offer even more advanced offline capabilities. This could include richer offline experiences, improved background synchronization, and better handling of intermittent connectivity.

2.7 Concepts from pwa

- Service Workers: Service workers are a crucial component of PWAs. They are JavaScript scripts that run in the background, separate from the web page, and can handle tasks such as caching, push notifications, and offline functionality. Service workers enable the PWA to work reliably, even with a poor or no network connection.
- web App Manifest: The web app manifest is a JSON file that provides metadata about the PWA, such as its name, description, icons, and the start URL. This file allows the browser to treat the web app as an installable application and enables features like adding to the home screen.
- Responsive Design: Responsive design ensures that the PWA adapts and looks good on various devices and screen sizes. CSS media queries and flexible layouts are commonly used to achieve responsive design.
- HTTPS: PWAs should be served over a secure HTTPS connection to ensure the integrity and security of data transmitted between the user and the server. This is a requirement for many PWA features, including service workers and push notifications.
- App Shell Architecture: The app shell is the minimal HTML, CSS, and JavaScript needed to render the basic user interface of the PWA. It is loaded first, providing a fast and reliable initial experience, and additional content is loaded dynamically as needed.
- Caching Strategies: Caching is a key aspect of PWAs, allowing them to work offline and load faster. Developers implement caching strategies using service workers to store and retrieve assets such as images, stylesheets, and scripts.
- Offline Capabilities: PWAs are designed to provide a seamless user experience even when offline or in poor network conditions. This is achieved through the use of service workers and caching strategies, allowing the app to function without an active internet connection.
- Push Notifications: Push notifications enable PWAs to send timely updates and engage users, even when the app is not open. Service workers are used to handle and display push notifications.
- IndexedDB and Local Storage: For storing data on the client side, PWAs leverage technologies like IndexedDB and local storage. These allow the app to persist data and provide offline functionality.
- Cross-Browser Compatibility: PWAs are designed to work across various browsers, and developers ensure compatibility by using standardized web APIs and features supported by most modern browsers.
- Automatic Updates: PWAs can update themselves automatically, ensuring that users always have the latest version without needing manual intervention. This is achieved through service workers and the use of a cache update strategy.

3 Specific Requirements for PWAs

3.1 External Interfaces

- Browser Compatibility: Ensure seamless functionality across major web browsers to guarantee a consistent user experience.
- API Integration: Implement interfaces with external services or APIs for enhanced functionality and data exchange.

3.2 Functions

- Offline Capabilities: Develop robust offline features utilizing service workers to enable users to access content without an internet connection.
- Push Notifications: Implement a system for timely push notifications to engage users and keep them informed.

3.3 Performance Requirements

- Loading Speed: Optimize performance for quick loading times, focusing on efficient data fetching and rendering.
- Responsiveness: Ensure a responsive design to provide a smooth user experience across various devices and screen sizes.

3.4 Design Constraints

- Security Measures: Adhere to HTTPS standards for secure data transmission, maintaining a secure connection.
- Browser Limitations: Consider and work around limitations imposed by different browsers to provide a consistent experience.

4 SOA (Service-Oriented Architecture)

Service-directed architecture (SOA) is a program development procedure that uses program elements called services to produce business applications.Developers use SOA to reuse services on different systems or combine several independent services to do complicated tasks. SOA is a type of architecture that allows applications to be built using a collection of services that communicate with each other. This communication can involve either simple data passing or it could involve two or more services coordinating some activity. Some means of connecting services to each other is needed.

4.1 advantages

SOA has the following advantages:

1. SOA is independent of vendors, products and technologies.
2. based on open standards.

3. based on the concept of loose coupling.
4. based on the concept of reuse.
5. based on the concept of interoperability.
6. based on the concept of modularity.

4.2 disadvantages

SOA has the following disadvantages:

1. SOA is complex.
2. SOA is expensive.
3. difficult to implement.
4. difficult to manage.
5. difficult to test.
6. difficult to secure.

5 Native app and Cross-platform

5.1 Native app

A native application is a software application that developers create specifically for a particular platform. Since developers design a native app for a specific device and its operating system, it can leverage the device's unique hardware and software features. Native apps can provide optimized performance and take advantage of the latest technology, such as a GPS, compared to web apps or mobile cloud apps developed to be generic across multiple systems.

Native app characteristics

- Native apps are developed for a specific platform, such as Android, iOS, Windows, or Blackberry.
- Native apps are written in a programming language specific to the platform they're being developed for.
- Native apps can take advantage of the latest technology available on mobile devices such as a global positioning system (GPS) and camera.
- Native apps can be downloaded from app stores (such as the Apple App Store or Google Play Store) and installed on mobile devices.
- Native apps can be used without an internet connection.

5.2 Cross-platform

Cross-platform software is a type of software application that is compatible with multiple operating systems. It is not specific to a single platform or device. Cross-platform software runs on multiple systems, such as Windows, Mac OS X, and Linux. Cross-platform software may also be referred to as multi-platform software or platform-independent software.

Cross-platform characteristics

- Cross-platform software is compatible with multiple operating systems.
- Cross-platform software can run on Windows, Mac OS X, and Linux.
- Cross-platform software may also be referred to as multi-platform software or platform-independent software.
- Cross-platform software can be written in a variety of programming languages.
- Cross-platform software can be downloaded from the internet and installed on a computer.

6 Appendices

Progressive Web Apps (PWAs) represent a type of application software delivered through the web. This section provides a detailed exploration of what defines a PWA. It also explains the benefits of PWAs and how they compare to other types of applications.

Explore the fundamental concepts associated with Progressive Web Apps, including their characteristics and advantages.