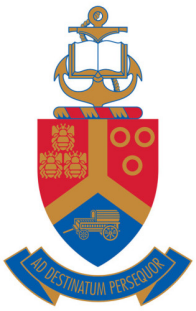

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science
Faculty of Engineering, Built Environment & IT
University of Pretoria

COS301 - Software Engineering

Software Requirements Specification

JavaScript

March 2022

Team Members:

Andrey	Omeltchenko	u04534205
Nasiphi	Mjobo	u18074074
Tshego	Manthata	u17110310
Kyle	Haarhoff	u19033347
Kajal	Bhoola	u20554240
Tlholo	Koma	u20575085
Matome	Makgopa	u18166793
Njabulo	Ntuli	u19062665

1 Introduction

1.1 Purpose:

The purpose of the following documentation is to provide the software requirements (functional and non functional), acceptance criteria, constraints and overview of the mobile feature (focusing on the functionality of a progressive web app) of the graduates portal.

1.2 Scope:

The following documentation includes the necessary information regarding the mobility of the graduates portal. The graduates portal is a portal where students and the industry come together. It is meant to be a platform that gives graduates and the industry a chance to find potential talent with the intent of getting job opportunities.

The JavaScript team has the task to develop a mobile feature for the graduates portal in order for its users to have a mobile experience. This feature ensures that all respective pages of the graduates portal website are mobile responsive and that the web app can work offline.

1.3 Acronyms, Abbreviations, Definitions:

- PWA - Progressive Web App : A progressive web app is a web app that uses service workers, manifests and certain web oriented features together with progressive enhancement to allow users to experience an experience on par with native apps.
- Service Worker : A JavaScript module that runs separately from the main browser thread, providing generic entry points for event-driven background processing.
- HTTPS - Hypertext Transfer Protocol Secure: A secure version of Hypertext Transfer Protocol (an application-layer protocol used for transferring hypermedia documentation, ie. HTML). It is used to send encrypted data between web browser and web servers.
- JSON - JavaScript Object Notation: A text format for storing and interchanging data.
- HTML - Hypertext Markup Language: Language in the form of code used to structure a web page together with its content
- CSS - Cascading Style Sheets: Language used to style HTML documentation and it adds description of how HTML elements should be displayed.
- URL - Uniform Resource Locator: A unique address of a resource on the Web.
- API - Application Programming Interface: Software that sends information interchangeably between a website (or app) and its user.
- UI - User Interface: the point where humans can interact and communicate with the computer- and its applications and websites (ie. display screen, keyboards, mouse , desktop presentation).
- GUI - Graphical User Interface: a type of user interface where users can interact with their computers through graphical icons and audio indicators.
- Native app : a software program that allows for mobility, it is developed for the use on a particular platform or device.

- Caching : a technique used to reduce network traffic rates by storing a copy of content into memory and serves it back upon request.

1.4 References:

- Afonso, F. (2020). Component Architecture : 3 Reasons to invest in one. Outsystems.Available from https://www.outsystems.com/blog/posts/component-architecture/ [accessed 16 April 2022].
- APImetrics.[nd]. Knowledge Base. APImetrics. Available from https://apimetrics.io/api-knowledge-base/apis-for-dummies/ [accessed 20 March 2022].
- AltexSoft (2020). The Good and the Bad of Angular Development. Available from https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/ [accessed 17 April 2022].
- Biorn-Hansen, A., Majchrzak, T.A., and Gronli, T.M. (2017) Progressive Web Apps: The Possible Web-native Unifier for Mobile Development. 13th International Conference on Web Information Systems and Technologies, 344-351. Available from https://www.scitepress.or g/papers/2017/63537/63537.pdf [accessed 12 March 2022].
- Bhoola, K., Koma, T., Haarhoff, K. Manthata, T., and Makgopa, J.(2022). PWA Offline Requirements. Available from https://docs.google.com/document/d/1IkTK_UsMMPmYwQtCZ-3Ga0Bk9zkTzxF2biEPL26P5rc/edit [accessed 17 April 2022].
- Google Developers. (2019) Lighthouse performance scoring. web.dev, 19 September 2019. Available from https://web.dev/performance-scoring/ [accessed 23 March 2022].
- Google Developers. (2020) What makes a good Progressive Web App? web.dev,, https://web.dev/pwa-checklist/ [accessed 23 March 2022].
- Jamadar, A. and Tandel, S.S. (2018) Impact of Progressive Web Apps on Web App Development. *International Journal of Innovative Research*

in Science Engineering and Technologyy, 7(9) 9439 - 9444. Available from https://www.researchgate.net/profile/Sayali-Tandel-2/ publication/ 330834334_Impact_of_Progressive_Web_Apps_on_Web_App_Development/links /5c5605d3a6fdccd6b5dde018/ Impact-of-Progressive-Web-Apps-on-Web-App-Development.pdf [accessed 09 March 2022].

- JSON org. [nd]. Introducing JSON. Available from https://www.json.org/json-en.html [accessed 20 March 2022].
- Kung, D.C. (2014) Object-Oriented Software Engineering: An Agile Unified Methodology. New York: McGraw Hill.
- Maddah-Ali, M.A and Niesen, U. (2012) Fundamental Limits of Caching. 1-19. Available from https://arxiv.org/pdf/1209.5807.pdf [accessed 21 March 2022].
- Malavolta, I., Procaccianti, G., Noorland, P. and Vukmirovic, P. (2017) Assessing the impact of Service Workers on the Energy Efficiency of Progressive Web Apps. *IEEE*, 1-11. Available from https://www.ivanomalavolta.com/files/papers/Mobilesoft_2017.pdf[accessed 21 March 2022].
- MDN Web Docs. (2022) Progressive web Apps. MDN. Available from https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps [accessed 09 March 2022].
- MDN Web Docs. (2022) HTML basics. MDN. Available from https://developer.mozilla.org/en -US/docs/Learn/Getting_started_with_the_web/HTML_basics [accessed 21 March 2022].
- MDN Web Docs. (2022) What is a URL? MDN. Available from https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_URL [accessed 22 March 2022].
- Myers, B.A. (2005) User Interface Software Tools. *ACM Transactions on Computer-Human Interaction Techtarget,* 2(1) 64-103. Available from

https://dl.acm.org/doi/pdf/10.1145/200968.200971?casa_token=UBWRF0TxHmUAAAAA:aYy-rNw6ARAKxHH62V7hiSA4PkdhLGLpkS2fkaXS82LfMBUutacG7yUt3kUOOkTNzS4XBn1-0C2n5g [accessed 21 March 2022].

- `Steiner, T. (2018) What is in a Web View? An Analysis of` Progressive Web App Features When the Means of Web Access is not a Web Browser. Developers Track, 23-27. Available from https://dl.acm.org/doi/pdf/10.1145/3184558.3188742 [accessed 11 March 2022].

1.5 **Outline of the rest of the SRS:**

The document follows the following scheme:

- A General Description
- Specific Requirements
- Acceptance Criteria

2 General Description

2.1 Context:

Mobile Feature:

The mobile feature has the general responsibility of a Progressive Web App (PWA) functionality. A Progressive Web App is a web app that is developed using a certain number of technical features together with standard patterns in order to achieve both web and native app features. The introduction of Web Apps has allowed users to view websites as an installed mobile app (particularly a mobile app on an android device).

A set of concepts together with keywords best describe Progressive Web Apps as: progressive, responsive, connectivity independent, app-like, fresh, safe , discoverable, re-engageable, installable and linkable. The mentioned attributes allow users to experience PWAs on par with mobile apps.

2.2 Progressive Web App Function:

The function of the progressive web app is to give the user a mobile experience of the graduates portal. The web app should function as follows:

- be responsive to a generic (ie.smartphone) device.
- be installable.
- work independent of internet connection (ie. work offline).
- have a high performance in terms of speed.
- responsive to smartphone device

2.2.1 Progressive Web App Structure

i. Encrypted HTTP protocol (HTTPS):

-A web app should be a secure site and should be browsed through a secure network for a Service Worker to register in the browser and act on events.

- This brings in the concept of a trusted website and thus securing the students information.

- the web app should allow the user to have access to the information they've entered in the graduates portal.

ii. Service Workers (Background script):

- A Service Worker is a background script written in JavaScript that handles and proxies all network connectivity, caching logic and background tasks.

- this allows the user to access the minimal static GUI and logic of the application without depending on internet connection, ie.the ability to use the application offline.

iii. Manifest File:

- A manifest file is a JSON text file which controls the way an app appears to the user.

-provides information about the application (ie. its name, author, icon and description) to the web app developers.

- the manifest can be used to change the behaviour and style of PWAs.

-provides users with quicker access and a richer experience.

iv. Application Shell (Core Architecture):

-The App Shell concept is concerned with loading a minimal user interface.

-it is defined as the minimal HTML, CSS and JavaScript powering a user interface.

-it is served up by the Service Worker and then the content is delivered.

-it is cached by the service worker from its source through API requests.

-it allows fast loading time together with cached and dynamic content.

2.3 User Characteristics:

The web app is intended for the use of:

1. Computer Science graduates from the University of Pretoria.
2. Companies looking to hire.

2.4 Constraints:

- All web app implementation and documentation should be done by the JavaScript team.
- The wireframe(s) design created by the team designer, should be the design the JavaScript team should follow.
- The Web app must be managed and maintained by the JavaScript team.
- The JavaScript team has to collaborate with UI Kit team in order to ensure the mobile friendliness of each web page.
- The UI Engineers must use the Angular framework and UI components created by the UI Kit team
- NestJS together with GraphQL must be used by the Service Engineer and API Engineer
- The tester uses CypressJs and Jest for testing purposes.

- The Data Engineer must use Prisma and Postgresql for building a database as well as database management.
- All work must be committed to GitHub.

2.5 Assumption and Dependencies:

The first version of the web app, will be an implementation of a basic web app. This version has the following features: working offline, installability, high performance, responsiveness to a generic device (in this case, a smartphone) and mobile friendliness.

For this version the web app, it is not the JavaScript team's intention to implement all of the respective PWA features (only the listed core features) as it is meant for a future release to get those features working.

3 Specific Requirements

3.1 Interface Requirements:

The web app must have a look that aligns with the UI of the graduates portal.

3.2 Functional Requirements:

FR1. Once the user opens the graduates portal, they must be able to install the web app and thereafter access it on their device.

FR1.1. The web app must have a web manifest with the correct fields.

FR1.2. The web app must work on localhost domain.

FR1.3. The web app must have an icon to represent it on the users device.

FR1.4. The web app must have a registered service worker

FR2. When the web app is opened the user must have an experience on par with a mobile app.

FR2.1. The web app must have a simple installation process.

FR3. The web app must be responsive to a generic device (ie. smartphone).

FR3.1. The web app must ensure that icons and images of the generic device of the appropriate size are retrieved.

FR4. The web app must work independent of internet connection (ie. work offline or at low network quality).

FR4.1. The user must be able to revisit pages they have once visited, even when internet connection is critical.

FR4.2. The web app must have a cached shell to load when offline.

FR5. All of the features of the web app should function as expected on a mobile app.

FR5.1. The web app should be mobile responsive.

3.3 Performance Requirements:

- The performance of the web app must be similar to that of a mobile app.
  - The time it takes from when the web app loads to the user being able to interact with it should be under 3.8 seconds
  - The time at which the first text or image is painted should be under 1.8 seconds
  - The render time of the largest image or text block visible within the view-port, relative to when the page first started loading should be under 2.5 seconds.
  - The total amount of time that the application's pages are blocked from responding to user input should be under 0.2 seconds.
  - The largest burst of layout shift of layout shift scores for every unexpected layout shift that occurs during the entire lifespan of a page should have a score of 0.1 or less.

3.4 Design Constraints:

The web app is dependent on the implementation of other feature teams, thus the JavaScript team cannot implement certain functions if certain feature teams have not implemented them yet.

3.5 Quality Requirements:

The web app should be linkable, the user should have the ability to link to the web app using a specific URL without the need for an app store or a complicated installation procedure.

3.6 Architectural Requirements:

3.6.1 Flexibility:

- The PWA must function on a generic mobile device (ie. smartphone).
- The PWA must work in all browsers (eg. Firefox and Google chrome).
- The PWA must be installable from a common variety of web browsers (ie. Firefox and Google Chrome).
    - The PWA must be usable on a basic level (ie. version 1.0) on older browsers.

3.6.2 Maintainability:

- There must be constant communication between developers of the PWA and the project owners in order to identify and repair errors thus improving the quality of application and user experience.
    - Clear and concise documentation on software requirements specification must be provided.
- The Angular framework is used to build the Graduates Portal and as the PWA uses a single codebase, in turn it must use the same framework.

    - Component-based architecture is more manageable and maintainable.
    - Components must be decoupled to aid more efficient maintenance and updates of code.
  - The PWA must contain a manifest file that must be kept up to date as it contains information about the website.
    - The manifest file of the PWA can be modified to reflect updates.
    - Updatable Web Manifest fields include:
      - Theme colour
      - Scope
      - Display
      - Icons
      - Shortcuts
    - The name or location of the PWA manifest file must not be tampered with, as this prevents the browser from updating the PWA.

3.6.3 Scalability:

- The server scalability is closely related to the scalability of the web app host server.
- The PWA caters for offline functionality once it has been installed, this in term increases the ability for users to use the PWA at the same time, as it is installed.
- The PWA must work anywhere from a single codebase as it is installable.
  - Single code base caters for wrapping up, which in turn provides high scalability.

3.6.4 Availability:

- The PWA must be available on any common web browser.

- The PWA must be available and easily accessed by any device, desktop or mobile.
- The PWA must be available at all times (day or night, connected to the internet or not) once it is installed.

3.6.5 Reliability:

- The installation of the PWA is reliable on internet connection.
- The PWA must have the capability of working offline, (i.e. independent of internet connection).
  - The service worker of the PWA must be responsible for caching content.
  - The PWA must use the application shell and IndexedDB to enable the web app to function regardless of internet connection.
- The PWA must ensure a stable experience to the user regardless of the quality of internet connection.

3.6.6 Usability:

- The PWA must use UI components developed by the UI kit team.
- The PWA must use a single codebase as the graduates portal, which is built on the Angular Framework which uses component-based architecture.
  - Components are self-sufficient and thus developers can reuse components throughout the the application (ie. the Graduates Portal, as a whole).
- The PWA must be mobile friendly.

3.6.7 Deployability:

- The PWA must be deployable to various platforms, such as:
  - Web application (ie. web browser)
  - Android

4 Acceptance criteria

4.1 When the user clicks on the install icon on the URL bar, they should be able to see the web app icon on the home screen of the respective device.

4.2 Upon opening the web app, the user should have the similar experience as they would if they were using a mobile app.

4.3 When the user once visited a page and losses connection or has bad internet connection, they still should be able to revisit the page they previously opened.

5 Appendices

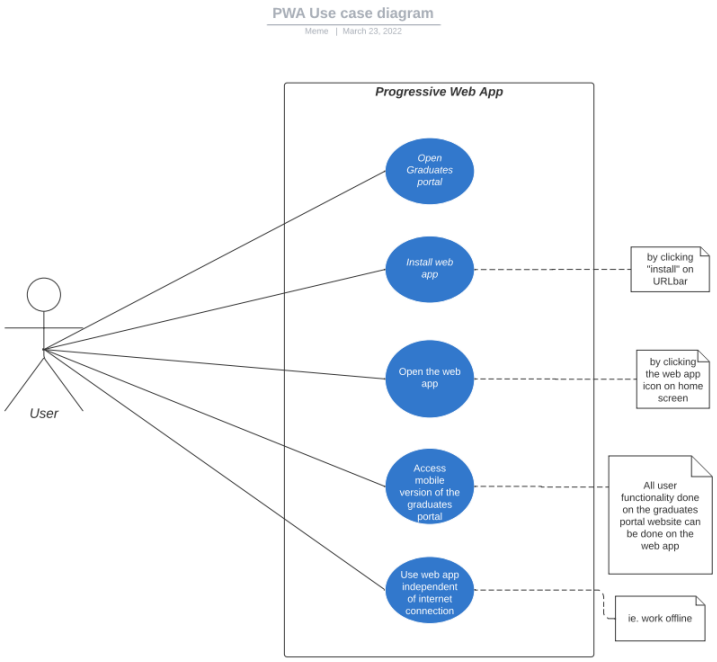

Figure 1: PWA Use-Case Diagram