Adobe & M1 Ltd.

M1 Limited

UI Review & Recommendations

TABLE OF CONTENTS

REVISIONS

VERSION	DATE	NAME	DESCRIPTION
1.0	22.09.2020	Mohamed Khalid	Initial draft version created.

1 Overview

This document contains a list of UI recommendations from the Adobe UI team on the M1 microsite implementation.

Few key points about the review and recommendations:

- This is a high-level quick review of UI implementation of M1 microsite
- Review is based on the workshop between Adobe and Infosys on 21-Sep-2020.
- Adobe UI team has not done a detailed code review of the M1 code base.
- Adobe team, both UI and AEM backend team, are not involved in day to day development process of M1 microsite.
- Review recommendations are based on the responses from Infosys team to the Adobe questionnaire.
- Adobe and Infosys also reviewed the product listing component during the workspace, code review is based on this review.

2 Recommendations

The following recommendations for improvement were identified during the review and are in high, medium, and low priority order.

RECOMMENDATION 1	PRIORITY: Medium
Category	Development Methodology
Issue/risk if not addressed	Mobile first layout is not followed, high chances of UI issues and consumes more time in development for desktop and mobile.
Recommendation(s)	Always we should follow mobile first layout development, so that we can design and develop all designs for both mobile and desktop screen. Having a mobile first approach for development would have avoided rework for many components.

RECOMMENDATION 2	PRIORITY: High
Category	Code Quality
Issue/risk if not addressed	SCSS files created but its fundamentals not used like Mixins, functions, variables instead they have used it like normal CSS and hardcoded colors, font-size and viewport sizes in all css files. If in future theming (ex: colors, fonts and font-sizes) to be changed across application then we need to rewrite all css files and endup with more efforts
Recommendation(s)	Always use variables for colors, fonts, viewport sizes and also use SCSS mixins which helps to reuse repetative CSS code. One place change will reflect in all SCSS files.

RECOMMENDATION 3	PRIORITY: High
Category	Code Quality
Issue/risk if not addressed	Some of the CSS properties are written inline in JS and HTML files. In future while updating CSS property if someone misses HTML syntax or JS syntax then entire application UI might break.
Recommendation(s)	Always write CSS code within SCSS/CSS file. Syntax miss will not break entire application UI instead it breaks particular component.

RECOMMENDATION 4	PRIORITY: Medium
Category	UI Architecture

Issue/risk if not addressed	Atomic design approach is not followed. All HTML templates are made static and those HTMLs cannot be reused in other places instead need to rewrite HTML and CSS again which consumes more effort and also increases the size of CSS file.
Recommendation(s)	Always break each section into individual component which can be reused across different pages/sections. Examples Accordions, Tabs.

RECOMMENDATION 5

PRIORITY: Medium

Category	Build Process
Issue/risk if not addressed	There is no separate UI Environment maintained for the project, UI team is currently coding within AEM component. This makes development more time consuming as each time after we do any change in HTML/SCSS/JS we need to run Maven build. Also, it consumes time in UI-AEM integration.
Recommendation(s)	Always we should maintain a separate UI Environment and should have build process like gulp, grunt or webpack. Once we code here and if component is working fine then final UI code can be integrated with AEM. This should have greatly improved productivity of UI team.

RECOMMENDATION 6

PRIORITY: Low

Category	UI Testing
Issue/risk if not addressed	As per the discussion, unit testing is done on browser simulators and simulators will not give exact results as physical devices. There will be more risk of UI breakage.

Recommendation(s)	If we don't have physical devices, then try using Browserstack.com which gives better results than browser simulators.

RECOMMENDATION 7

PRIORITY: Low

Category	Security
Issue/risk if not addressed	Cookie values are directly exposed. We are able to see the saved cookie objects on browser developer tool and read it clearly
Recommendation(s)	Always Cookie/Session data to be encoded and then it should be saved. We can use methods like encodeURIComponent() decodeURIComponent()