Sprint 4 Report

Front-End – Responsive Web Page Creation

Sprint 3 & 4 : Création d'une page web responsive

Réalisé par :
Karrouach Ansar

Encadré par :
Hamza Bahlaouane
Abdelmajid Bendrif

11 février 2026

VOID

Table des matières

1 Project Setup – Vite and Tailwind CSS 2

2 Responsive Header and Navigation 2

3 Dark Mode Implementation 3

4 Typography and Google Fonts 4

5 Responsive Form with Validation 5

6 FAQ Interactive Accordion 6

7 Statistics Section – Animated Counters 7

8 Conclusion 8

1. Project Setup – Vite and Tailwind CSS

Objective : Initialize a modern front-end project using Vite and configure Tailwind CSS for responsive styling.

1. Create Vite project :

```
npm create vite@latest my-project
cd my-project
```

2. Install Tailwind :

```
npm install tailwindcss @tailwindcss/vite
```

3. Configure plugin in vite.config.js :

```
import { defineConfig } from 'vite'
import tailwindcss from '@tailwindcss/vite'

export default defineConfig({
  plugins: [tailwindcss()],
})
```

4. Import Tailwind in CSS :

```
@import "tailwindcss";
```

5. Start development server :

```
npm run dev
```

Result : Tailwind successfully integrated with Vite. Project ready for responsive development.

2. Responsive Header and Navigation

Objective : Create a fully responsive header with navigation, CTA button, dark mode toggle, and mobile burger menu.

Features implemented :

- Semantic HTML structure
- Responsive navigation (desktop + mobile)
- SVG icons integration
- Sticky header

- Mobile dropdown menu

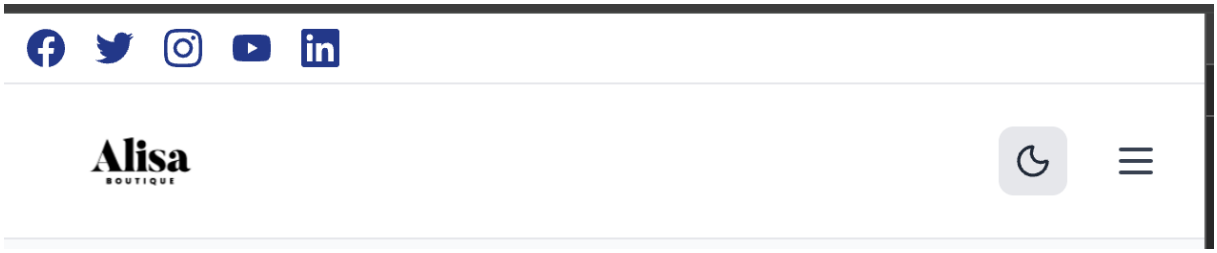

FIGURE 1 – Responsive Header – Mobile View

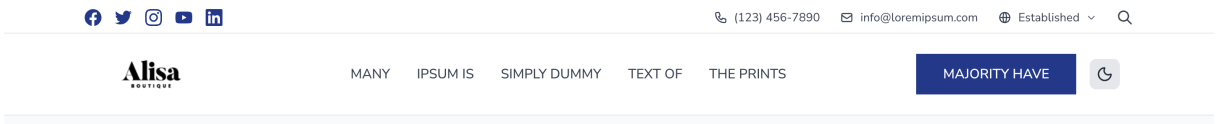

FIGURE 2 – Responsive Header – Desktop View

Result : Header adapts correctly to different screen sizes using Tailwind breakpoints.

3. Dark Mode Implementation

Objective : Implement a dark/light theme toggle using native JavaScript and Tailwind custom variant.

- Custom dark variant in CSS
- Theme persistence using localStorage
- Flash prevention on page load
- Dynamic icon switching

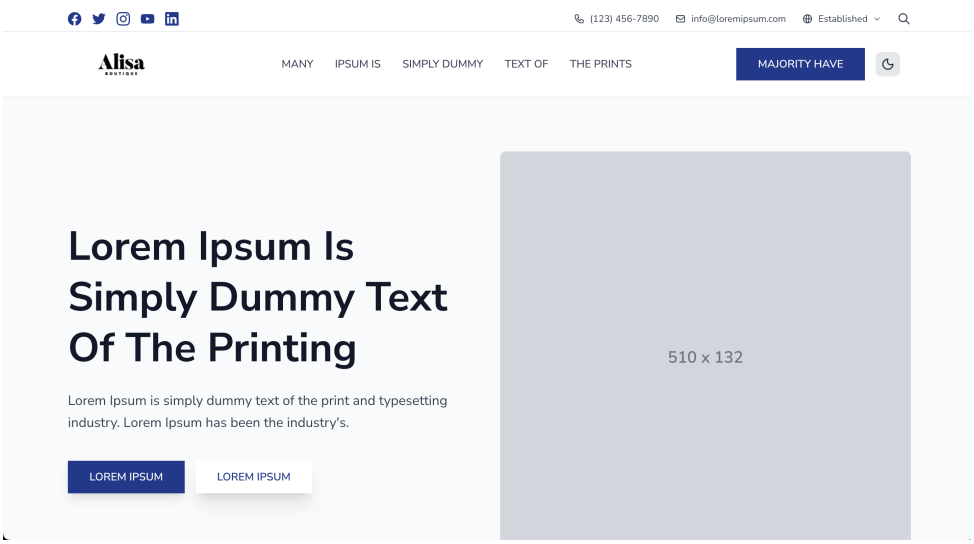

FIGURE 3 – Light Mode Interface

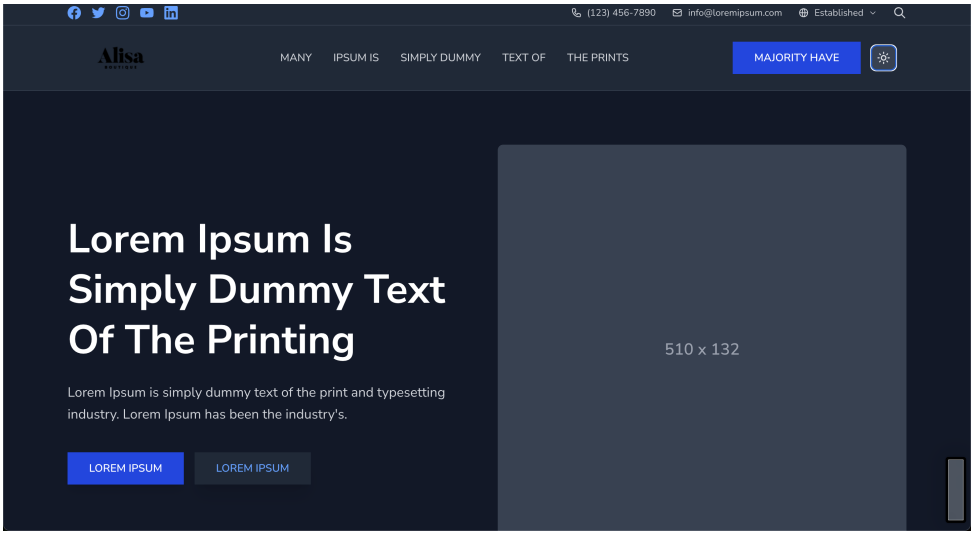

FIGURE 4 – Dark Mode Interface

Result : Smooth dark mode toggle with persistent user preference.

4. Typography and Google Fonts

Objective : Integrate custom Google Font for improved UI design.

```
<link href="https://fonts.googleapis.com/css2?family=Poppins
:wght@300;400;500;600;700&display=swap" rel="stylesheet">
```

```
font-family: 'Poppins', system-ui, Avenir, Helvetica, Arial, sans-serif;
```

Result : Modern typography integrated across the website.

5. Responsive Form with Validation

Objective : Create a responsive form with client-side validation using native JavaScript.

Features :

- Responsive grid layout
- Email validation (regex)
- Moroccan phone number validation
- Error handling with dynamic messages
- Success message and form reset

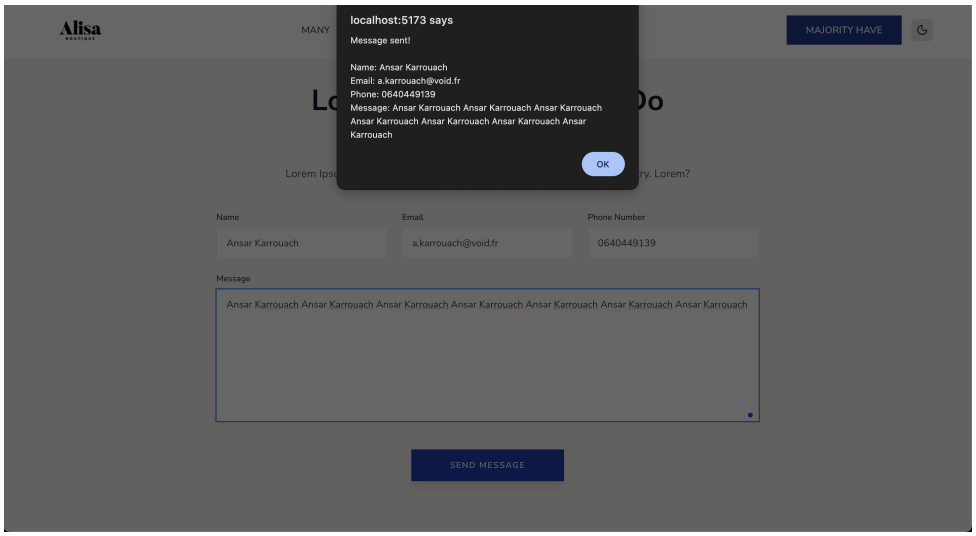

FIGURE 5 – Form – Valid Submission

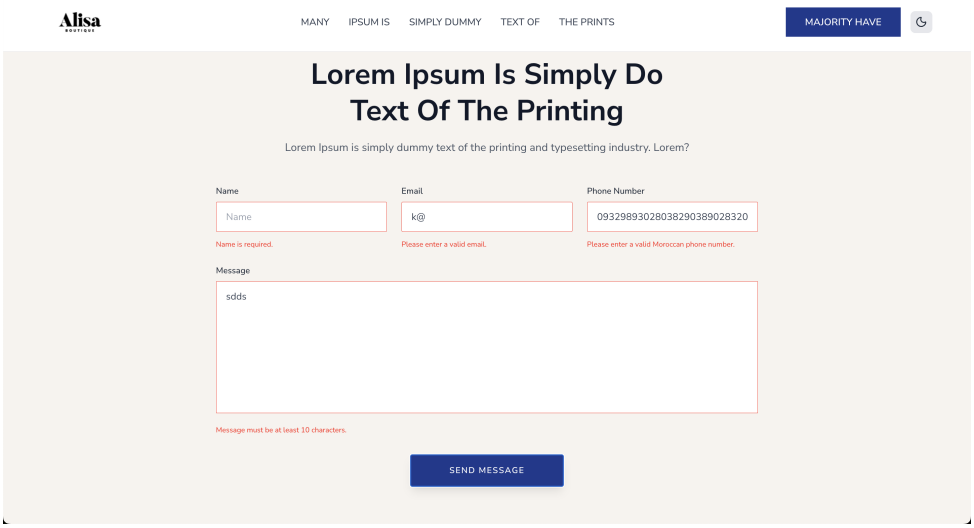

FIGURE 6 – Form – Validation Errors Displayed

Result : Fully functional responsive form with real-time validation feedback.

6. FAQ Interactive Accordion

Objective : Implement interactive FAQ accordion using native JavaScript.

- Toggle open/close behavior
- Auto-close other items
- Dynamic icon switching (+ / -)
- Responsive design

FIGURE 7 – Interactive FAQ Section

Result : FAQ section works dynamically without any framework.

7. Statistics Section – Animated Counters

Objective : Create animated statistics using Intersection Observer and native JavaScript.

Features :

- Count-up animation
- Intersection Observer trigger
- One-time animation execution
- Responsive layout (grid system)

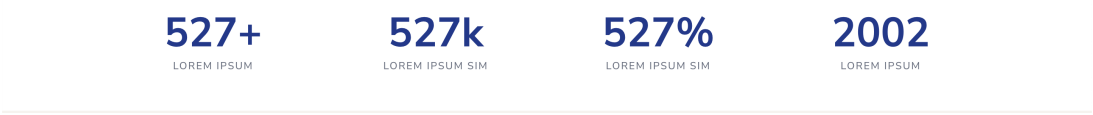

FIGURE 8 – Statistics Count-Up Animation

Result : Animated counters triggered when section enters viewport.

8. Conclusion

Sprint 4 Progress Summary

During this sprint, the following components were successfully implemented :

- Vite + Tailwind setup
- Fully responsive header and navigation
- Dark mode toggle with persistence
- Google Fonts integration
- Responsive form with validation
- Interactive FAQ accordion
- Animated statistics section

The project follows semantic HTML structure, responsive design principles, and native JavaScript interactions as required.

Remaining advanced features (RTL, performance optimization, Lighthouse tests, responsive images with `<picture>`, video embed, etc.) will be implemented in the next phase.