User Interface

General look of the web app, components, images, layout and how it compares with the Invision mock up

SCORE: 4.5 / 5

Comments

- General look and feel of the website very close to Invision with minor differences with good attention to detail
    - Grey border on the bottom of the top nav bar and around input fields darker than Invision
    - Disabled add new phone number button still has hover over mouse icon rather than default
    - Footer text positioning in centre rather than on the right
- Text, images and buttons well handled with buttons functional where relevant (linking to contact/about)
- Carousel component, next/previous arrows and pagination largely matching Invision
- Aligning, spacing and padding of panels, inputs, images and text well handled with minor differences
    - Padding on some elements inside panels on homepage slightly less than Invision

---

Responsiveness

How well the web app, images and components behave and look across varying screen sizes (1440px down to 320px) along with if any functionality is affected in any manner

SCORE: 4.8 / 5

Comments

- Exceptional responsiveness, image handling and look on mobile is excellent
- Form, home page panels, layout and text flawlessly handled from desktop to tablet to mobile
- Carousel functional on all screen sizes with pagination and excellent image handling
- Only improvement I could suggest is to centralise the wrapped text/button element inside the panels on images as soon as tablet screen size is rather than later

---

Code Quality

How well structured are the folders & files, the code itself, componentisation, JS & React best practices, HTML+CSS/SCSS quality, npm package usage (formik/axios vs manual/fetch), Redux (if used) and handling of API calls

SCORE: 3.25 / 5

Comments

- General layout very simple and easy to follow albeit lacking componentisation

    - Only carried out for Carousel component, Scroll to top (which is much nicer when implemented with hooks) and carousel image. All other files are for individual pages and the nav/footer.
- Carousel component fetching images itself rather than dedicating it to another isolated file with a single function responsible for fetching images and handling response, parsing and returning relevant data.
- Fetch (native API) used instead of axios (npm package), missing out on bunch of functionality and response handling regarding HTTP headers and JSON parsing, leaving it to be manual
- No TypeScript, SCSS or Formik
    - All styles for all components in a single style.css file rather than along with the relevant components
- For the code written and componentisation done it is generally well handled with slightly out of convention naming handling
- Good use of hooks where used

Functionality

Are all features of the website functional and the quality of the functionality on desktop and mobile: Navigation, Form handling (submission & errors), carousel on home page, form add new phone number + add address details handling

SCORE: 4.5 / 5

Comments

- All functionality well handled; Navigation, buttons, Carousel, form behaviour with possible improvement:
    - On form submission no need to submit empty address details object when user has not selected
    - Remove button for optional extra phone number field
- Carousel component flawlessly handled working well across all screen sizes and drag to swipe functionality included
- Form error handling well done and navigation to success screen on submission
    - No need to make API request when validation errors can be caught on front end (empty fields)

Summary

General summary of the above highlighting areas of good performance and areas requiring improvement

Exceptional styling and responsiveness, good attention to detail with minor deviances from the Invision. Form component, carousel component and images flawlessly handled across all screen sizes.

Areas to improve

- Use TypeScript, SCSS and Formik to facilitate developing and maintaining scalable applications

- Make use of available npm packages to speed up, ease and improve quality of development: axios, redux, formik, SCSS and react wrapper for Swiper rather than Swiper core
- Isolate API calls into own files and functions to keep view components responsible only for UI related functions
- Use more componentisation to generate reusable components