SPECS - Rocket Builder

Vers 0.1 - 2023-06-08

#1 - Solution

The product needs to empower users to build websites visually. Here is the primary use case:

- Framework: VanillaJS
- Design and Components pulled from a Backend via a simple API
- The user selects the layout
    - Website (layout) suitable for presentation sites
    - Dashboard Layout
- Select the design
    - Each type of layout will have designs associated. Here are some samples
    - **Website** (layout)
        - Material KIT: https://demos.creative-tim.com/material-kit/index.html
        - Soft Design: https://demos.creative-tim.com/soft-ui-design-system/index.html
    - **Dashboard** Layout
        - Material Dashboard: https://demos.creative-tim.com/material-dashboard/pages/dashboard.html
        - Soft Dashboard: https://demos.creative-tim.com/soft-ui-dashboard/pages/dashboard.html
- Once a project type is selected and the DESIGN the user can do the following
    - Create/Delete Pages
    - Edit pages by dragging & drop components
    - Edit component properties like texts, images, and links (internal, and external links)
- Allowed actions of Projects:
    - Create
    - Delete
    - Save on disk (not database)
    - Preview
    - Download
- Builder layout
    - Left: components grouped by type (collapsable):
        - Footers
        - Navigation bars
        - Sidebars
        - Pricing
        - Team Cards

        - HERO Section
        - Image Carousels
    - Center: built website or dashboard
    - Right: Active component properties
        - Once a component is selected, the right column displays the props and allows the user to edit

#2 - Repository

https://github.com/app-generator/rocket-builder

#3 - Sprint #1

The first version of the product needs to provide a minimal codebase able to do the following:

- Simple Backend Server that provides:
    - List of the UI KITS
    - Components & Layout (master page) for each UI KIT
- Builder powered by VanillaJS
    - UI Container: Mantis MUI (free product)
- Builder Layout
    - Navigation bar (for builder controls) 100% width
    - Left Panel: 15% width, for Kits & Components management
    - CENTER (active builder area) - 70% width
    - Right Panel: 15% width, used for the
- Layout Navigation bar
    - This control empowers the user to:
        - Give a name to the project
        - See the current design name
        - Preview the project
        - Export the project
- LEFT panel
    - Components list that the user can drag & drop
- CENTER - Active builder for the current page
    - The User can drop the components
    - For each component
        - Edit texts (in place)
        - Edit IMG (external URL)
        - Change order (up & down)
        - Delete Component
- RIGHT

- When a component or element is selected, this panel shows the associated properties

Just to summarize this sprint:
- The user can select a UI KIT (based on the information pulled from the API)
- The user can create pages
- The user can edit an existing page
- The user can add/remove or edit components for a page
- The user can preview the site
- The user can download the site (HTML format)
    - The export needs to be in ZIP and should include theme assets and all the design work provided by the user

#3.1 NPM Package

This section explains how the product can be used as an NPM package or directly CDN

#3.1.1 Library INIT

This section explains how to use the library via CDN and the initialization parameters.

#3.2 Codebase Footprint

SRC directory
- **constants** directory
  All constants should stored be here.
- **Utiles** directory
  api directory
    componentApi.ts
    …

  utiles.ts
- **Config** directory
  https://prnt.sc/FEYMuT7qsf9i
- **Functions** directory
  All functions should separate by each feature
  Example:
  downloadFeature.ts
  addTabFeature.ts
  dndFeature.ts
  …
- **Styles** directory

All styles should stored be here

- Index.ts

  index.js file should play a crucial role as the entry point of the application