# Solution Architecture

* Problem statement
* Solution overview
* Architecture
* Non-functional requirements
* Build & Release
* Tech stack

## Problem Statement

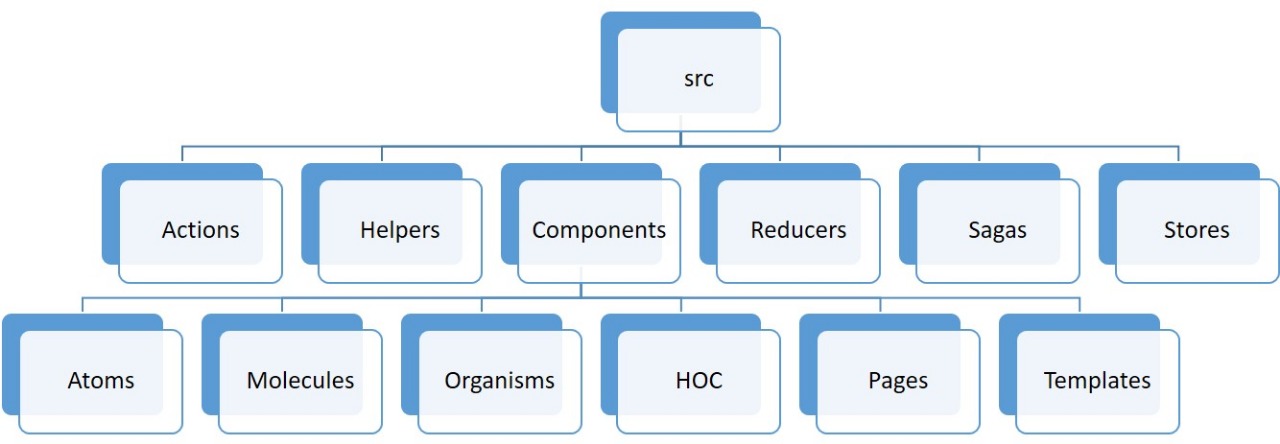
Create a hacker news clone where users should be able to view the news feed, be able to comment, up-vote or hide any of the feed items and pagination. The solution should be responsive in nature for tablet & mobile devices and also perform well on non-functional requirements such as SEO, performance and accessibility.

## Solution Overview

## Platform Principles

#### Atomic Design Methodology

Follow Atomic design methodology for the folder structure to get the benefits of component’s reusability, maintainability and for easy updates. Following is the folder structure:



Re-usable Components-Develop components that can be re-used for different components.

Mobile First approach- Take Mobile First responsive approach for development

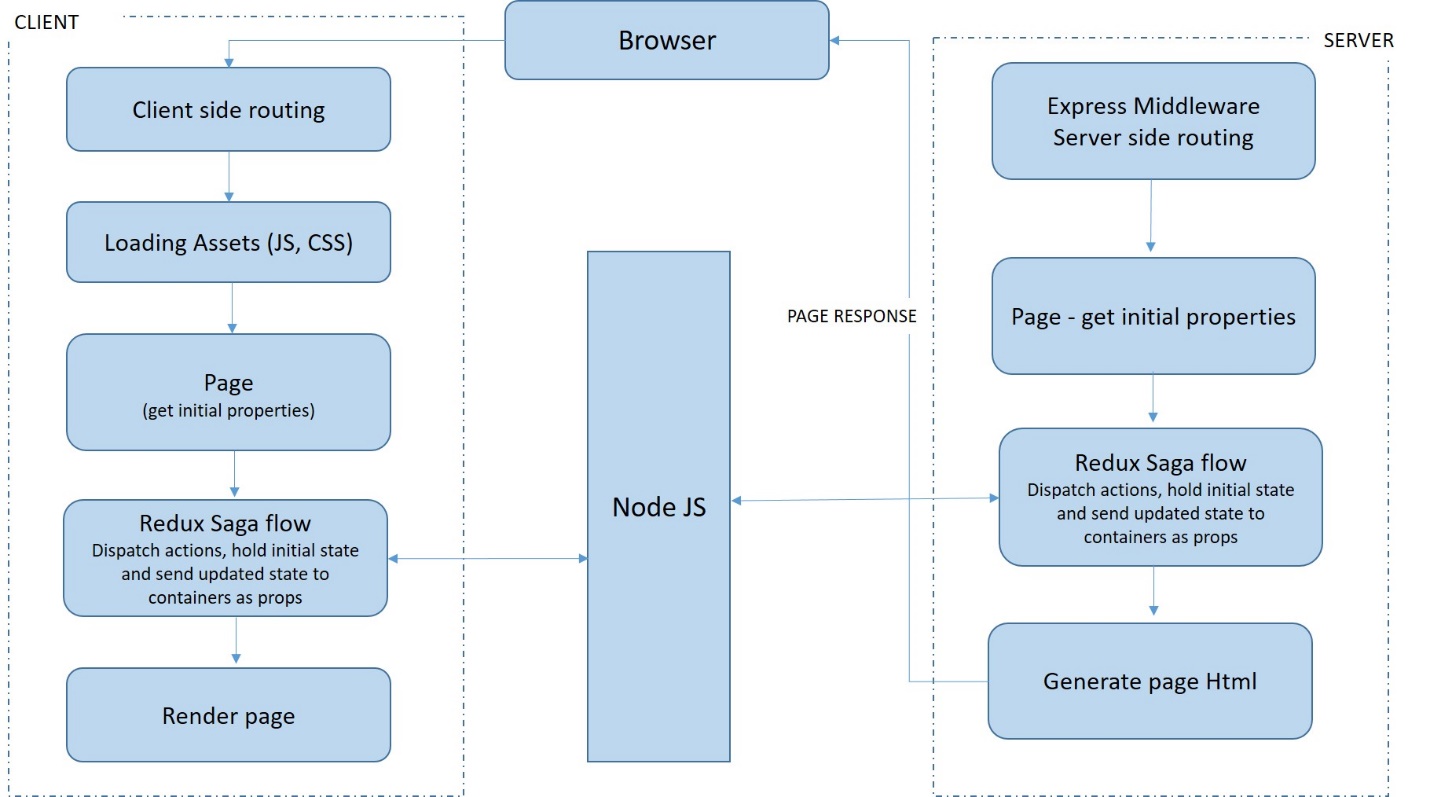
Scalability & Maintainability- Architecture will be scalable to enable future addition of new components or creating variations of existing components. High Order components will enable to keep code more maintainable and shareable.

Pure Components – Pure component is one of the most significant ways to optimize React applications. The usage of Pure Component gives a considerable increase in performance because it reduces the number of render operation in the application.

#### High level solution approach

1. Considering the highly dynamic nature of the application where the DOM content needs to be refreshed frequently to achieve features such as hide/show of a particular feed item, up-votes, multi tab view, paginate through the list of feed items etc., requires a performant framework such as ReactJS which provides features such as Virtual DOM, Pure, High order and reusable components etc. and can be utilized for this purpose
2. State management will be handled using Redux. The state of an application is kept in a store and each component can access any state that it needs from this store. 3 main parts of Redux are Actions for dispatching events; Reducers for returning the updated state & Store for holding the application state. To implement features such as up-vote (there is no direct integration required at this point) and hide, the store will be maintained in local storage
3. Since we need to integrate with the APIs (to retrieve the feed details in this case), redux-saga which is a redux middleware library can be utilized to handle asynchronous calls and potential failures much more effectively. It achieves this by leveraging an ES6 feature called Generators, which allows to write asynchronous code, and is very easy to test and debug redux app with “redux-devtools-extension”
4. To make the application more SEO friendly, it is imperative to use server side rendering to return the complete HTML response so that the crawlers do not see blank content when accessed. Node + Express JS will be used for this purpose
5. **Responsive web design –** Custom CSS will be used for responsive web design keeping the application size in mind. Using SCSS for CSS preprocessing
6. **Code quality**
   1. ES linting will be used for JS code quality
   2. Style lint will be used for JS code quality
   3. Snapshot and functional testing with Jest and Enzyme

## High level Architecture

****

## Non Functional Requirements

### Performance

* Implementing PWA, Service worker and manifest
* Chunking through Webpack
* Code minification for react and the content being processed and sent by Node
* Prefetch/Preload – to load assets and prioritized the rendering

### Accessibility

AA accessibility to be supported and so the pages will be designed accordingly

### SEO

All the basic SEO guidelines needs to be followed as part of the implementation such as

* Title Tags
* Meta Descriptions
* Heading tags
* Image Alt texts
* Descriptive text for Links

Additionally, Server side rendering using Express JS will be implemented to allow search engines to index the content similar to static sites than JS rendering most of the content.

Tools such as Screaming Frog and Lighthouse can be used to assess the compliance with SEO best practices throughout the development life cycle.

## Caching Strategy

A cache is a component that transparently stores data so that future requests can be served more quickly. It is safe to cache data that doesn’t cache need to be recomputed often, whereas fetching new data would result in duplicated data. Depending upon the volatility of the content, we can focus our efforts on storing cached content in long-term storage.

## Security

### Preventing XSS Vulnerability

* + Query parameters must be fixed so that Viewer cannot change it and hack the site
  + Origin should be set as same Origin
  + Use “noopener noreferrer” in rel attribute while using Target as \_blank
  + Run Npm Audit for Npm packages
  + Scan website for vulnerabilities
  + Use a secure host

## Build & Release

**Travis CI** will be used a tool of choice for CI/CD Integration. When a build is run, Travis clones the linked repository into its virtual environment and perform a series of specified steps and can also deploy the code to the hosting platform including sending any required notifications.

The instructions will be added to the .travis.yml which will be placed at the root of the repository. Travis supports man programming languages, and we are using node JS as the language specifier for our build process. The core instruction for Travis is to kick off the Webpack build which further performs following steps:

1. Install all the required dependencies
2. Code quality checks will be performed using SASS Linting and ES Linting
3. Once the quality checks are passed, all the required JS and CSS assets will be bundled

## Tech stack

* Node: Server, routing
* Express JS: Server side rendering
* React: Isomorphic support, faster view update and rendering
* REDUX: State management
* Saga: Asynchronous calls and side defect management
* SCSS: CSS processor
* Axios: API calls
* Webpack 4 (API): React apps build and deployment
* Jest and Enzyme: For unit tests, code coverage and sonar reporting
* Eslint: To ensure the JS code quality
* Stylelint: To ensure the CSS code quality
* Babel: ES6 to ES5 compilation