**Overview:**

The purpose of this application is a blog style photography site. Registered users will be able to post an image with small text caption. Other registered users will be able to view, like, and comment on these posts.

Nonregistered users will only be able to view posts. Homepage will show the initial 16 articles with the newest post appearing first. Subsequent API calls to server can retrieve an additional 6 posts at a time.

Registered users will have access to a dashboard that will display their avatar, username (if applicable), posts, profile, and setting.

**Technologies:**

* React.js – To build out the user interface into a series of components.
* Redux.js – Global state management. Allows various React components to update and receive updates as necessary depending on the current state. Redux will be used to store the loaded posts in state where React can access it and display the posts.
* Bootstrap 4 – CSS Framework to help build out UI. Use customized CSS as necessary.
* Node.js – JavaScript runtime environment. JavaScript code is converted into faster machine code. Node.js uses an event-driven non-blocking I/O model.
* Express.js – Node.js framework to assist with route handling, integrating MongoDB database, handling and requests.
* MongoDB – Nonrelational database used to store data as documents in collections. Saves user and post collections.
* Mongoose.js – MongoDB object data modeling library. Represents data as JSON objects then maps them over the underlying database. Mongoose Model 🡪 uses schema 🡪 to create new documents 🡪 gets placed into collection.
* JSON Web Token – Used for authentication.
* Bcrypt.js – Used to hash passwords.

**User Interface:**

Styling will be influenced by [www.speedhunters.com](http://www.speedhunters.com) for displaying posts on the homepage.



* Articles will show a spinner as the post is being loaded.
* Top Header – Site name and some background styling.
* Navbar – Home / Categories / Contributor List / Login / Register
* React will send a request to the server to return the latest 15 posts. Posts will be saved into Redux state so that we don’t have to make an additional query request.
* See more post links to Posts page listing ALL post. Add pagination as a feature later on.
* Will have a “Load More” button to query an additional six posts at a time. Load button changes to spinner while querying.
  + Load more will fire off a redux action. Pass in 15 + number of queries (saved in state).
  + Action will hit backend API endpoint to retrieve 6 posts from server.
  + 6 posts will be inserted into a new component. Will be appended to React.
  + Have place holder conditional for the new posts. Use a .map to render each component.
* Below that will be a footer.

