Would you rather

# **Project Instructions**

1) Use React to build your application’s UI. Remember that composition is key. It’s rarely a mistake to break a component into smaller pieces. Look for opportunities to reuse your components.

2) We recommend using [Create React App](https://github.com/facebookincubator/create-react-app) to generate your submission since it's the easiest way to ensure you have everything the project reviewer will need to install and run your app.

3) By walking through the Planning Stage and the Coding Stage of the Chirper Project, we’ve given you a useful template for building Redux applications. We recommend using this template for building your “Would You Rather?” Project. Remember that planning your project and its architecture before starting to code will save you a lot of debugging time later on!

4) Use Redux to manage your application state. For this application, most of the application’s state should be managed by Redux. You may use component state to handle form input fields and controlled components. Otherwise, the rest of the state for your application should be controlled by your reducers.

5) While the focus (and specification) of this project is based on functionality rather than styling, please ensure that your app is presentable and easy to navigate.

6) Please carefully test your app against the [rubric](#Rubric) (check rubric on next page) to make sure *all* of the rubric requirements are met. Your project must meet all of the rubric criteria in order to pass.

## **PROJECT SPECIFICATION / RUBRIC**

**Would You Rather?**

**Application Setup**

|  |  |
| --- | --- |
| CRITERIA | MEETS SPECIFICATIONS |
| Is the application easy to install and start? | The application requires only npm install and npm start to install and launch. |
| Does the application include README with clear installation and launch instructions? | A README is included with the project. The README includes a description and clear instructions for installing and launching the project. |

**Login Flow**

|  |  |
| --- | --- |
| CRITERIA | MEETS SPECIFICATIONS |
| Does the application have a way to log in and log out?  Does the application work correctly regardless of which person the user impersonates? | 1. There should be a way for the user to impersonate/ log in as an existing user. (This could be as simple as having a login box that appears at the root of the application. The user could then select a name from the list of existing users.)  2. The application works correctly regardless of which user is selected.  3. The application allows the user to log out and log back in. The user should be logged in to submit new polling questions, vote, and view the leaderboard.  4. Once the user logs in, the home page is shown.  5. Whenever the user types something in the address bar, the user is asked to log in before the requested page is shown. |

**Application Functionality**

|  |  |
| --- | --- |
| CRITERIA | MEETS SPECIFICATIONS |
| Does the home page have the desired functionality? | 1. The answered and unanswered polls are both available at the root.  2. The user can alternate between viewing answered and unanswered polls.  3. The unanswered questions are shown by default.  4. The name of the logged in user is visible on the page.  5. The user can navigate to the leaderboard.  6. The user can navigate to the form that allows the user to create a new poll. |
| Are the polling questions listed in the correct category (Unanswered vs Answered), and do they have the desired functionality on the home page? | Each polling question resides in the correct category. For example, if a question hasn’t been answered by the current user, it should be in the “Unanswered” category.  A polling question links to details of that poll.  The polls in both categories are arranged from the most recently created (top) to the least recently created (bottom). |
| Are the details of each poll displayed with all of the relevant information? | 1. The details of the poll are available at questions/:question\_id.  2. When a poll is clicked on the home page, the following is shown:  1. the text “Would You Rather”;  2. the picture of the user who posted the polling question; and  3. the two options.  3. For answered polls, each of the two options contains the following:  1. the text of the option;  2. the number of people who voted for that option;  3. the percentage of people who voted for that option.  4. The option selected by the logged in user should be clearly marked.  5. When the user is logged in, the details of the poll are shown. If the user is logged out, he/she is asked to log in before before being able to access the poll.  6. The application asks the user to sign in and shows a 404 page if that poll does not exist. (In other words, if a user creates a poll and then the same or another user tries to access that poll by its url, the user should be asked to sign in and then be shown a 404 page. Please keep in mind that new polls will not be accessible at their url because of the way the backend is set up in this application.) |
| Does the voting mechanism work correctly? | 1. Upon voting in a poll, all of the information of the answered poll is displayed.  2. The user’s response is recorded and is clearly visible on the poll details page.  3. When the user comes back to the home page, the polling question appears in the “Answered” column.  4. The voting mechanism works correctly, and the data on the leaderboard changes appropriately. |
| Can users add new polls? | 1. The form is available at/add.  2. The application shows the text “Would You Rather” and has a form for creating two options.  3. Upon submitting the form, a new poll is created and the user is taken to the home page.  4. The new polling question appears in the correct category on the home page. |
| Does the leaderboard work correctly and have the desired functionality? | 1. The Leaderboard is available at/leaderboard.  2. Each entry on the leaderboard contains the following:  1. the user’s name;  2. the user’s picture;  3. the number of questions the user asked; and  4. the number of questions the user answered.  3. Users are ordered in descending order based on the sum of the number of questions they’ve answered and the number of questions they’ve asked. |
| Is the application navigable? | The app contains a navigation bar that is visible on all of the pages.  The user can navigate between the page for creating new polls, and the leaderboard page, and the home page without typing the address into the address bar. |
| Does the application interact with the backend correctly? | The data that’s initially displayed is populated correctly from the backend.  Each user’s answer and each new poll is correctly recorded on the backend. |
| Does the code run without errors? Is the code free of warnings that resulted from not following the best practices listed in the documentation, such as using key for list items? Is the code formatted properly? | The code runs without errors. There are no warnings that resulted from not following the best practices listed in the documentation, such as using key for list items. All code is functional and formatted properly. |

**Architecture**

|  |  |
| --- | --- |
| CRITERIA | MEETS SPECIFICATIONS |
| Does the store serve as the application’s single source of truth? | The store is the application’s source of truth.  Components read the necessary state from the store; they do not have their own versions of the same state.  There are no direct API calls in the components' lifecycle methods. |
| Is application state managed by Redux? | Most application state is managed by the Redux store. State-based props are mapped from the store rather than stored as component state.  Form inputs and controlled components may have some state handled by the component. |
| Does application state update correctly? | Updates are triggered by dispatching action creators to reducers.  Reducers and actions are written properly and correctly return updated state to the store. |
| Does the architecture of the application make sense? | The code is structured and organized in a logical way.  Components are modular and reusable. |

### **Suggestions to Make Your Project Stand Out!**

· Add the functionality for creating new users.

· Add authentication.

· Add a loading bar.

# ***Template for Building React/ Redux Projects***

## **Planning Stage 📐**

### **Step 1 - Draw All of the Views of the App**

We need to determine the look and functionality of each view in your app. One of the best approaches is to draw each view of the app on paper so that you'll have a good idea of what information and data you're planning to have on each page.

Instead of paper and pencil, you can be a bit more digital and use [software for creating mockups](https://codingsans.com/blog/mockup-tools). If you were given project specifications, check your mock against them to make sure that you have all of the required features.

### **Step 2 - Break Each View Into a Hierarchy of Components**

For this step,

* draw boxes around every component; and
* arrange the components into a hierarchy

### **Step 3 - Determine What Events Happen in the App**

We need to take a look at what is happening in each component. Let's determine what actions the app or the user is performing on the data. Is the data being set, modified, or deleted?...then we'll need an action to keep track of that event!

### **Step 4 - Determine What Data Lives in the Store**

Remember that the main problems that Redux (and the react-redux bindings!) was meant to solve were:

* Propagation of props through the entire component tree.
* Ensuring consistency and predictability of the state across the app.

According to Dan Abramov, the creator of Redux, we should follow the following principle for determining whether to store a piece of data in the store or in a React component:

*"Use Redux for state that matters globally or is mutated in complex ways… The rule of thumb is: do whatever is less awkward."*

Take a look at [Organizing State](https://redux.js.org/faq/organizing-state) and [How to choose between Redux's store and React's state?](https://github.com/reactjs/redux/issues/1287) for further information about this.

**Coding Stage**

**Step 1** - Design the [shape of the state](https://redux.js.org/recipes/structuringreducers/normalizingstateshape) and [create reducers](https://redux.js.org/basics/reducers).

**Step 2** - Create a [Redux store](https://redux.js.org/api/store). Connect logger middleware (optional) and Redux Thunk middleware (alternatively, you can use Redux Saga, etc.).

**Step 3** - For each view that needs access to the store, create the component and connect it to the store.

**Step 4** - For the component you created in the last step, create actions and action creators. Check that everything works correctly.

**Step 5** - Repeat Step 3 & Step 4 for each component that needs access to the store.

**Step 6** - Create presentational components and confirm that everything works correctly.

**Step 7** - Add React Router.

**Step 8** - Add finishing touches and make sure the project meets the [rubric](#Rubric).

Remember, this is just a template. As you build more projects, you'll modify this template to suit your needs. You may also find it more intuitive to use a different approach. Regardless of the approach you take, however, **planning out your app is imperative to success**.

**Before submitting, please verify that your project:**

1. Meets specification according to the project [**rubric**](#Rubric).
2. Includes all of the files necessary to install and launch your web application on a local web server. For files that include JSX, please refrain from using the .jsx extension (you can prefer .js). You can assume that your reviewer will have npm installed on their machine.
3. Adheres to udaicity [**HTML**](http://udacity.github.io/frontend-nanodegree-styleguide/index.html), [**CSS**](http://udacity.github.io/frontend-nanodegree-styleguide/css.html), [**JavaScript**](http://udacity.github.io/frontend-nanodegree-styleguide/javascript.html), and [**Git**](https://udacity.github.io/git-styleguide/)style guidelines.

**Project Start files can be downloaded from**

<https://github.com/aamirpinger/reactnd-project-would-you-rather-starter>