Dokumentation Erstellen einer Cocktail App

Paper Prototyp / Requirements

Der folgende Prototyp wurde mithilfe von Powerpoint erstellt. Er dient lediglich als erster Anhaltspunkt für das Erstellen der Cocktail App. Die Elemente der Prototypen orientieren sich an Material – UI, einer Dokumentation für React Elemente.

Dieses Bild entspricht der Home-Seite der App. Über das Zutaten Eingabe Feld können eingekaufte Zutaten hinzugefügt werden. Diese werden bei der Rezeptsuche für verfügbare Cocktails berücksichtigt.

Das Feld „Do you want to go shopping?“ kann über die Check Box ausgewählt werden, wenn der Benutzer vor der Zubereitung seiner Cocktails noch einkaufen gehen möchte. Dann werden ihm auch Rezepte angezeigt, für die nicht alle Zutaten vorrätig sind.

In der unteren Hälfte können über die Check Boxen vorhandene Zutaten aus dem System entfernt werden. Dies sollte geschehen, wenn eine Zutat verbraucht worden ist.

Über den rechten Pfeil gelangt man auf die nächste Seite, auf der die verschiedenen Rezepte angezeigt werden.

-> Der rechte Pfeil wurde durch ein kleines Cocktail Symbol ersetzt. Die Funktionalität bleibt jedoch wie oben beschrieben.

In dieser Liste werden die verfügbaren Rezepte angezeigt. Der farbige Punkt steht für die bereits vorhandenen Zutaten.

Der grüne Punkt zeigt an, dass alle benötigten Zutaten bereits vorhanden sind. Sie können direkt loslegen!

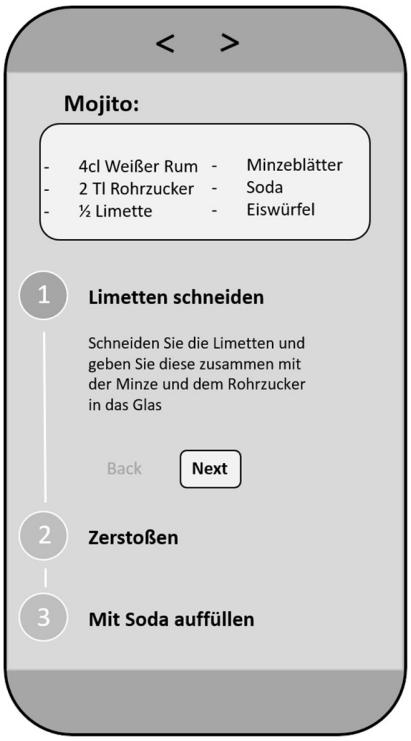
Der gelbe Punkt zeigt an, dass mindestens 50% der benötigten Zutaten vorhanden sind. Die restlichen Zutaten müssen noch besorgt werden.

Der rote Punkt zeigt an, dass weniger als 50% der benötigten Zutaten vorhanden sind. Für diese Cocktails müssen noch einige Sachen besorgt werden.

Nachdem sich der Benutzer für einen Cocktail entschieden hat, landet er auf der Zubereitungsseite.

Der Pfeil oben links bringt dich zurück auf die Startseite, falls Anpassungen getätigt werden müssen.

-> Anstelle der Punkte wurde das gesamte Feld in der entsprechenden Farbe gefärbt.

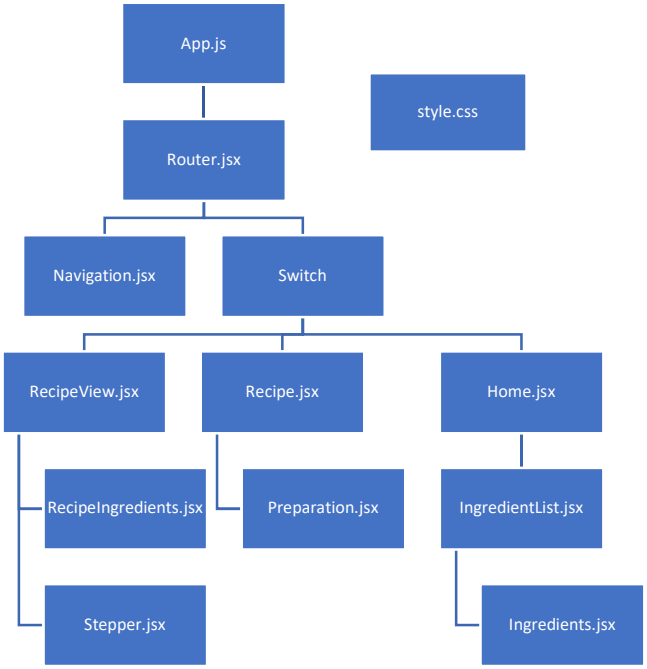


Auf dieser Seite erhält man einen detaillierten Überblick über die Menge der benötigten Zutaten.

Über einen sogenannten Stepper können die verschiedenen Zubereitungsschritte abgearbeitet werden. Sobald der erste Schritt abgearbeitet ist, kann über den „Next“ Button der nächste Zubereitungsschritt abgerufen werden.

Möchte man einen Schritt zurück springen, gelangt man über den „Back“ Button auf den vorherigen Zubereitungsschritt. Am Ende kann über das Feld „Back to the beginning“ von vorne angefangen werden.

Klassendiagramm



Der Aufbau der React.js App ist auf der linken Seite zu sehen. Die App.js beinhaltet einen Router.jsx, in dem der React Router definiert wird und durch den Switch wird ein wechsel zwischen den Seiten RecipeView.jsx, Recipe.jsx und Home.jsx ermöglicht.

- Über die RecipeView.jsx wird die Datei RecipeIngredients.jsx und die Stepper.jsx aufgerufen. Dadurch wird sowohl die Zutatenliste, als auch die Zubereitungsschritte dargestellt.

- Von Recipe.jsx wird die Datei Preparation.jsx aufgerufen, in der die Rezpte mit den verfügbaren Zutaten gematcht und in den entsprechenden Farben dargestellt werden.

Die Navigation.jsx ist über alle Seiten des Routers verfügbar.

Konzept

Über die Datei Home.jsx erfolgt die Eingabe der Zutaten. Eingegebene strings werden in der Funktion AllIngredients gespeichert und mithilfe von Props können die eigegebenen strings an Komponenten übergeben werden. Innerhalb der Funktion können diese Properties wie Variablen verwendet werden. Die eingegebenen Informationen werden über die Funktion IngredientList an die Datei Ingredientlist.jsx übergeben. Mithilfe der Funktion Ingredients.jsx können verfügbare Zutaten aus dem Bestand entfernt werden. Über Ingrdientlist.jsx wird immer der aktuelle Bestand festgehalten.

In der Recipe.jsx wird auf eine Firebase Echtzeitdatenbank zugegriffen in der die Datei recipes.json gespeichert ist. In dieser Datei sind die Rezepte mit allen Mengenangaben, Zubereitungsschritten und benötigten Zutaten gespeichert. Da die Rezeptliste ständig aktualisiert werden kann, wurde hier auf eine state Membervariable zugegriffen. Ändert sich der State innerhalb der Komponente, wird automatisch die Render Funktion neu aufgerufen und so kann die Rezeptliste stets up to date gahlten werden.
In der Datei Preparation.jsx wird über die Funktion Prepare überprüft ob Cocktailrezepte mit den verfügbaren Zutaten matchen und es wird die matchingRate berechnet. Diese matching Rate wird dann wie im Prototyp beschrieben visuell mithilfe der Farben grün, gelb und rot dargestellt.

Über die Datei RecipeView.jsx wird die Zubereitungsansicht des Cocktails dargestellt. Dazu wird aus der Firebase Datenbank die Datei recipes.json ausgelesen und den Bestandteilen werden entsprechende Varaiblen zugewiesen.
In der RecipeView.jsx werden die Funktionen RecipeIngredients und Stepper aufgerufen. Die Funktion RecipeIngredients ist in der Datei RecipeIngredients.jsx gespeichert und gibt eine Liste mit den Zutaten

und den entsprechenden Mengen eines Cocktails an.
Die Funktion Stepper ist in der Datei Stepper.jsx gespeichert und gibt einen vertikalen Stepper der Schritt für Schritt abgearbeitet werden kann aus. Dieser Stepper wurde mithilfe von Material UI erstellt und zieht die Daten ebenfalls aus der recipes.json.

Neben der Material UI Library wurde axios verwendet um asynchrone HTTP-Anfragen an REST Endpunkte zu senden und so geringere Wartezeiten bei Datenbankabfragen zu erreichen.

Abschluss / Fazit

Bei dem Erstellen der Cocktail App war der Gebrauch von Material UI sehr hilfreich. Mithilfe der React Library konnten vorgefertigte Elemente benutzt werden und dann in der eigenen App entsprechend angepasst werden.
Es war jedoch nicht immer eindeutig, welche Abschnitte des Code's verwendet werden mussten und was angepasst werden musste. Dennoch konnten dadurch auch aufwendige Elemente wie der Stepper mehr oder weniger einfach in die App integriert werden.
Durch die vorgefertigten Pakete war es jedoch schwierig die Elemente über css nach eigenem Bedarf zu verändern. Viele Einstellungen waren vorgefertigt und ein einfaches ändern der Farbe stellt sich bei bestimmten Elementen als nahezu unmöglich dar.

Ebenfalls musste ich feststellen, das die Sachen, die in einem Prototyp schnell gezeichnet sind, sich in der Umsetzung doch schwieriger als gedacht darstellen. Daher wurden einige Funktionen in der richtigen App anders dargestellt als im Prototyp eingezeichnet.

Nachdem die Kinderkrankheiten beim Programmieren allerdings überwunden worden sind, und man vertrauter mit React wurde hat es eigentlich ziemlich gut funktioniert und man erhält eine für mobile Endgeräte optimierte App mit vielen Funktionen.

Die Änderungen der React App wurden während der Erstellung regelmäßig auf GitHub gepusht. Über folgendem Link kann auf das öffentliche Repository zugegriffen werden:
https://github.com/peterbehrens/dhbwWS20

Matrikelnummer: 8747156

Auszüge aus der fertigen App