Prototyping Projektdokumentation

Name: Marc Schwendemann

E-Mail: schma@students.zhaw.ch

URL der deployten Anwendung: https://cookr-site.netlify.app/

Einleitung

Cookr ist eine Website, um Kochrezepte zu verwalten und schnell Gerichte zu finden, die man zubereiten kann mit den Zutaten, die man gerade zur Verfügung hat.

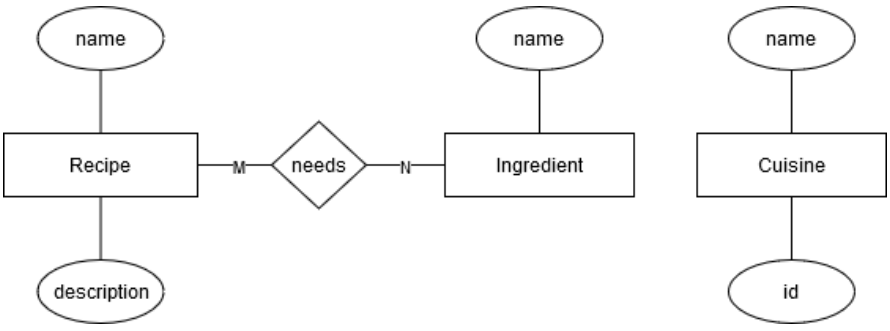
Die Website verfügt über eine Page, die alle in der Datenbank gespeicherten Rezepte benutzerfreundlich darstellt. Über diese Seite oder mit der passenden URL kann die Detailansicht eines Rezeptes aufgerufen werden, wo alle Informationen (Name, Beschreibung, Zutaten) über das Rezept ersichtlich sind. Von dort aus kann ein Rezept sowohl angepasst als auch wieder gelöscht werden. Ebenfalls können neue Rezepte erstellt werden, was wie das Anpassen mit einem Formular umgesetzt wurde. Dabei werden die Eingaben validiert, sodass sichergestellt ist, dass Name und Beschreibung eingegeben, sowie mindestens eine Zutat ausgewählt wurde.

Die Kernfunktion der App ist die Möglichkeit, die Anzeige der Rezepte zu filtern, sodass nur solche angezeigt werden, die Zutaten benötigen, die man zur Verfügung hat. Dazu kann auf der Übersichtsseite der Rezepte der Filter angezeigt werden, der aus Checkboxen aller Zutaten besteht. Die Zutaten können selektiert werden, woraufhin nur noch Rezepte angezeigt werden, die Zutaten enthalten, welche aktiviert wurden. Mit diesem Feature kann man ganz einfach schauen, welche Zutaten man zuhause hat und welche Gerichte man damit kochen könnte.

Die Website verfügt über eine Ansicht, um zu sehen, welche Zutaten in wie vielen Rezepten vorkommt. Diese Information ist in einem Balkendiagramm dargestellt, sodass der Benutzer schnell sehen kann, wie wichtig und beliebt welche Zutaten sind.

Ein weiteres Feature ist die Page zum Verwalten der Lieblingsküchen (Italienisch, Indisch, etc.). Dort sind alle in der Datenbank vorhandenen Küchen aufgelistet, angeordnet in einer Rangliste, bei dem der Index jeder Küche als Attribut auf der Entität hinterlegt ist. Die Elemente können durch eine Library per Drag and Drop in der Liste verschoben werden. Wenn man zufrieden ist mit der Sortierung, kann die Rangliste gespeichert werden, wodurch der Index jeder Küche in der Datenbank angepasst wird.

Datenmodell



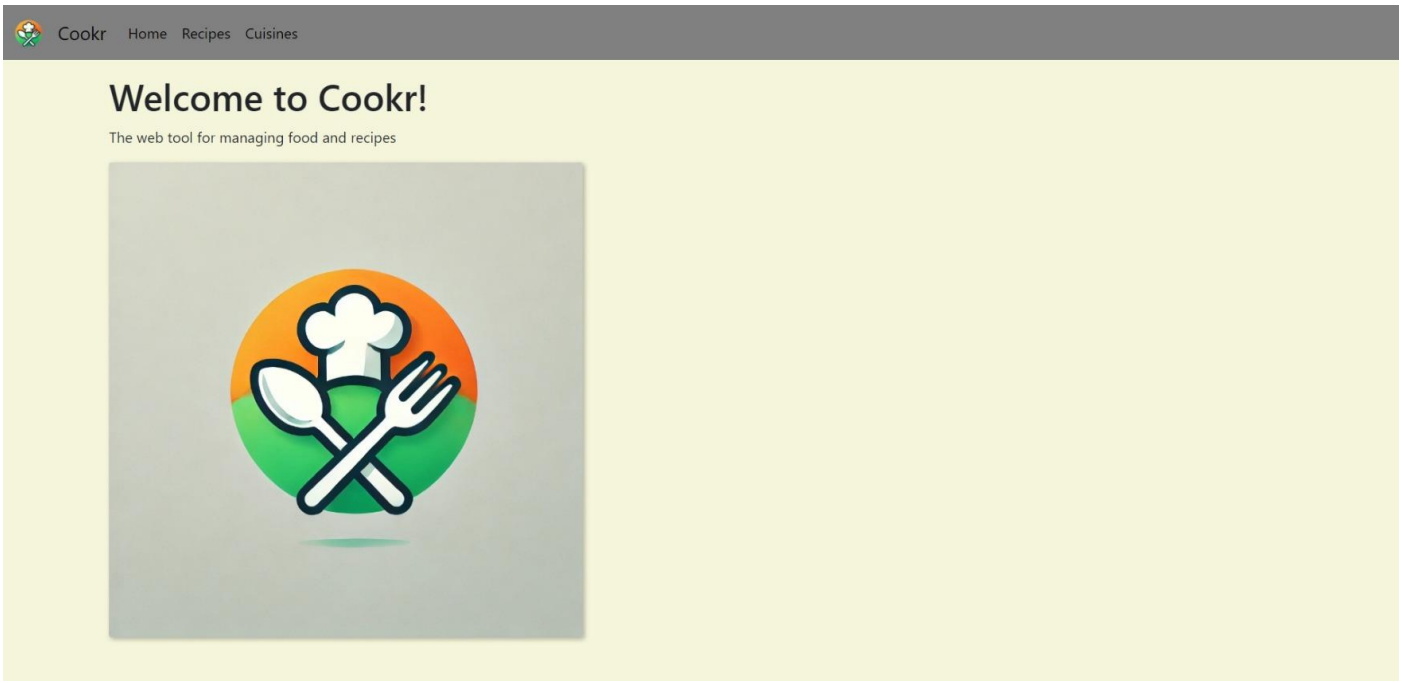
Die beiden Entitäten *Recipe* und *Ingredient* und deren Beziehung sind selbsterklärend. Ein Rezept hat ein Name und eine Beschreibung sowie eine beliebige Anzahl and Zutaten. Eine Zutat hat einen Namen und kann bei beliebig vielen Rezepten vorhanden sein.

Eine *Cuisine* hat einen Namen und das Attribut *id.* Die *id* ist nicht der Identifikator der Entität, das wäre in MongoDB das Attribut *_id*, sondern dient als Nummer für die Position in der Rangliste. Die Entitäten, die aus der Datenbank geladen werden, sind sortiert nach dieser Nummer.

Beschreibung der Anwendung

Homepage

Route: /



Die Startseite ist schlicht gestaltet. Ein Willkommenstext, eine kurze Beschreibung der Website und das Logo.

Dateien

- routes/+page.svelte

Navbar

So sieht die Navigationsleiste beim ersten Aufruf aus. Das Logo, *Cookr* und *Home* führen zur Startseite (/). *Recipes* und *Cuisines* routen zu /recipes bzw. /cuisines.

Die Navbar ist mittels Bootstraping responsive. Sobald das Fenster zu klein ist, um alle Elemente anzuzeigen, verschwinden *Home, Recipes* und *Cuisines.* Stattdessen erscheint am rechten Ende ein Button, bei dessen Klick die 3 Elemente darunter angezeigt werden.

Die aktuelle Seite wird in der Navigationsleiste orange markiert und wenn man mit der Maus über ein Element hovered, wird dieses weiss.

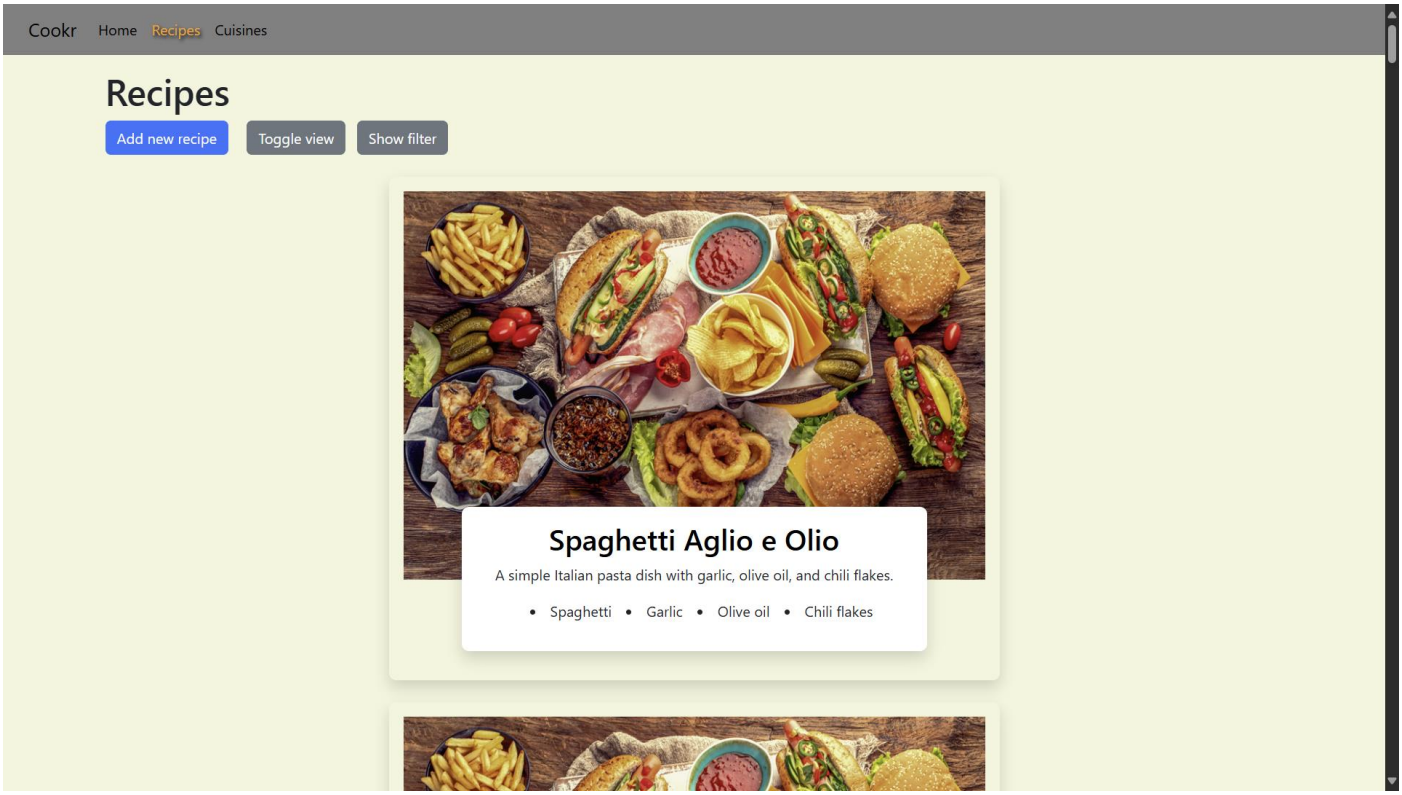
Dateien

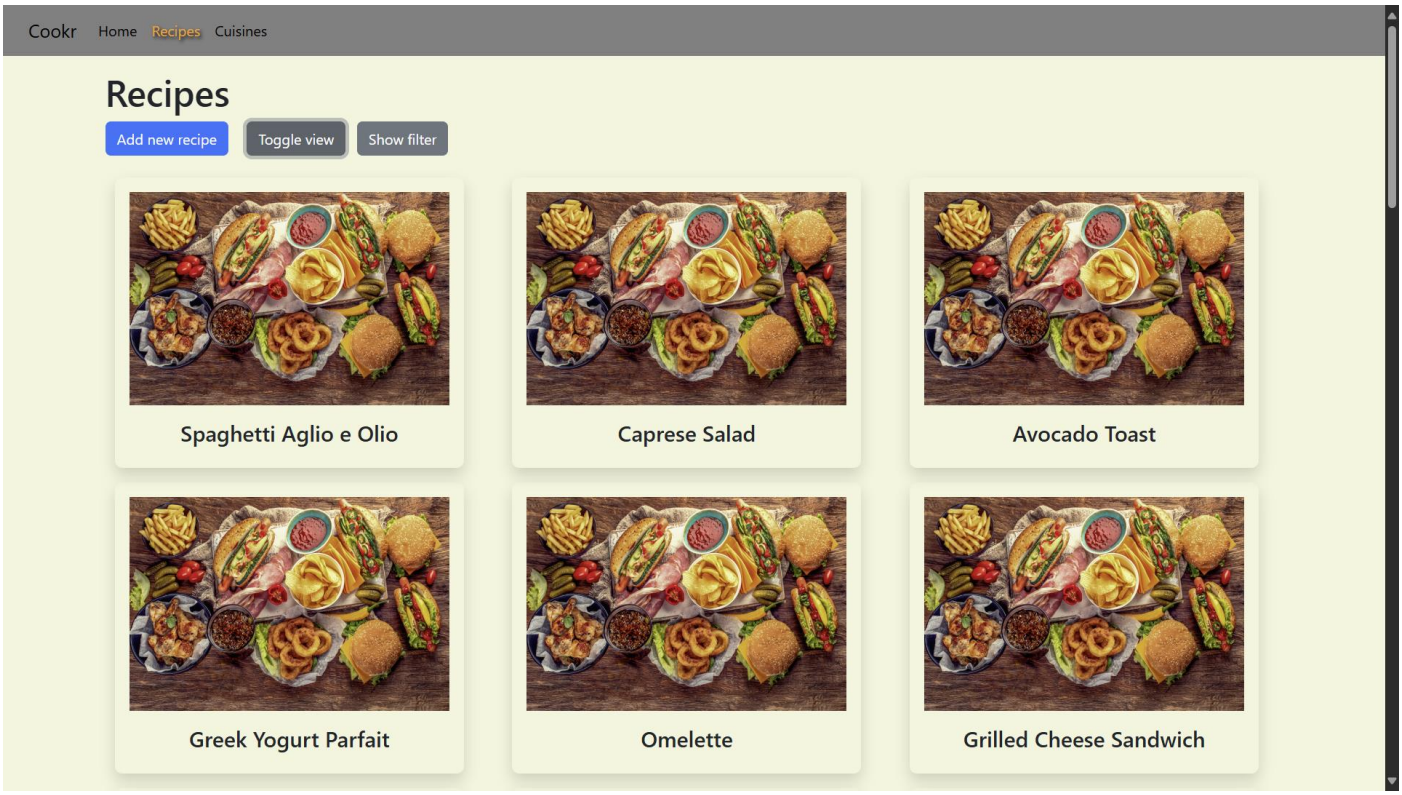
- routes/+layout.svelte

Übersicht Rezepte

Route: /recipes

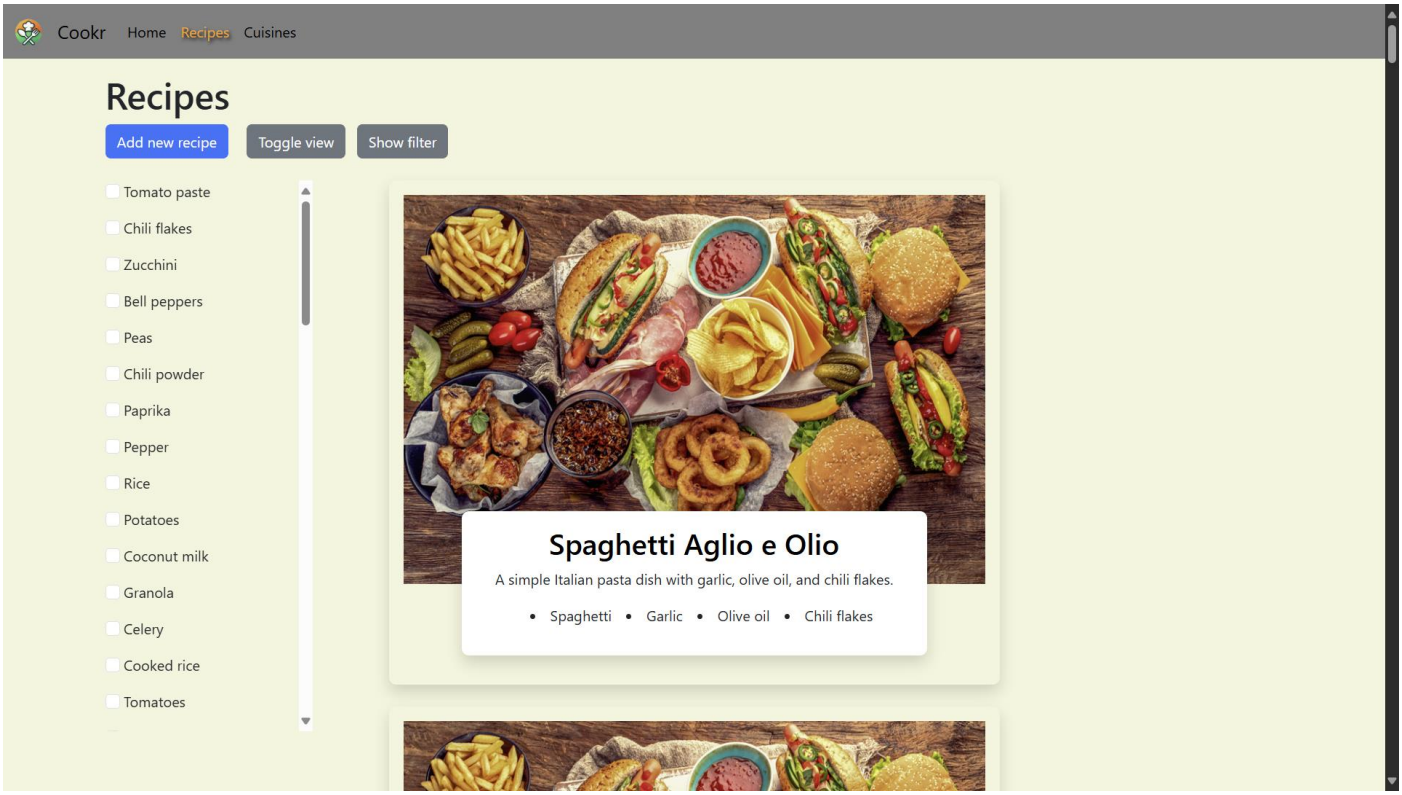
Auf der *Recipes*-Seite werden alle Rezepte angezeigt, die von der Datenbank geladen werden.

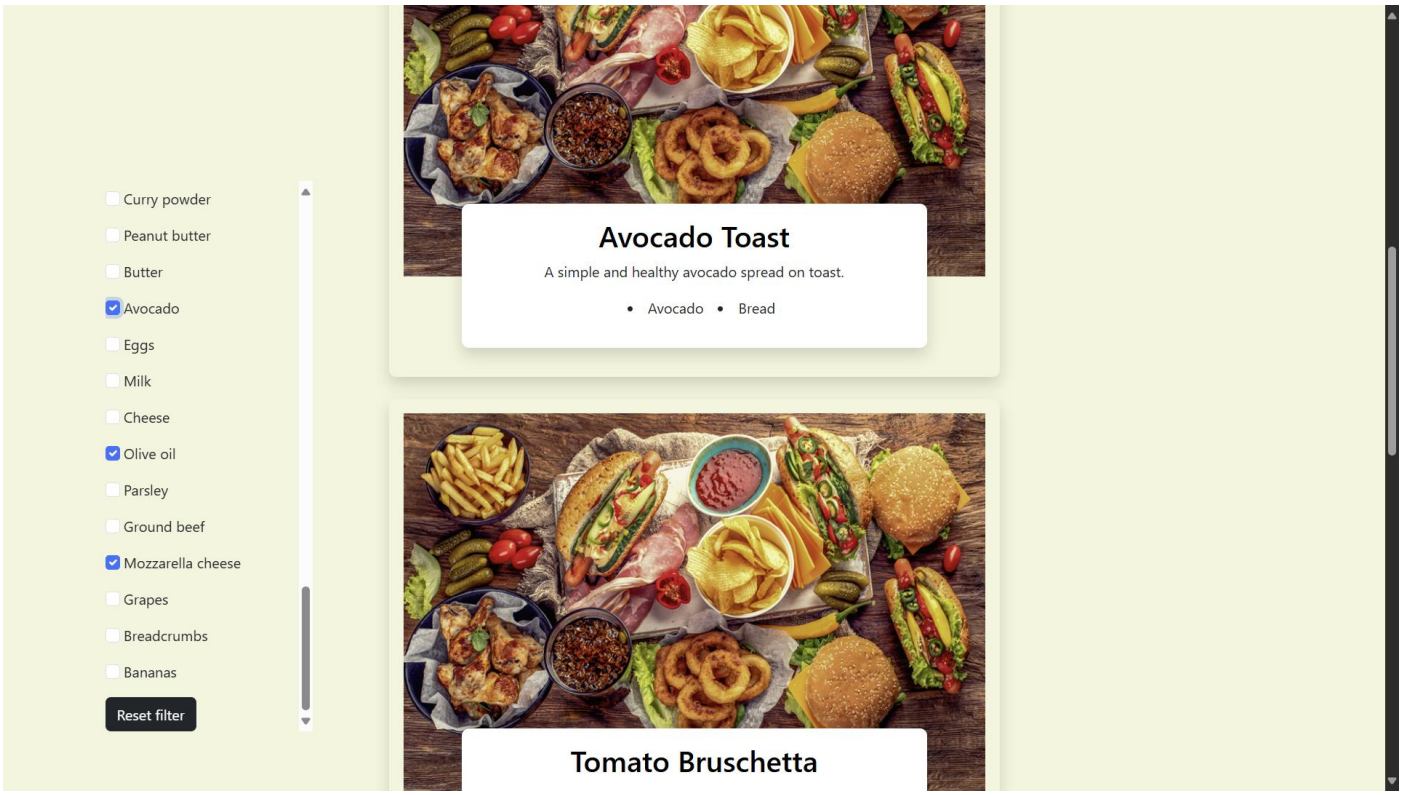


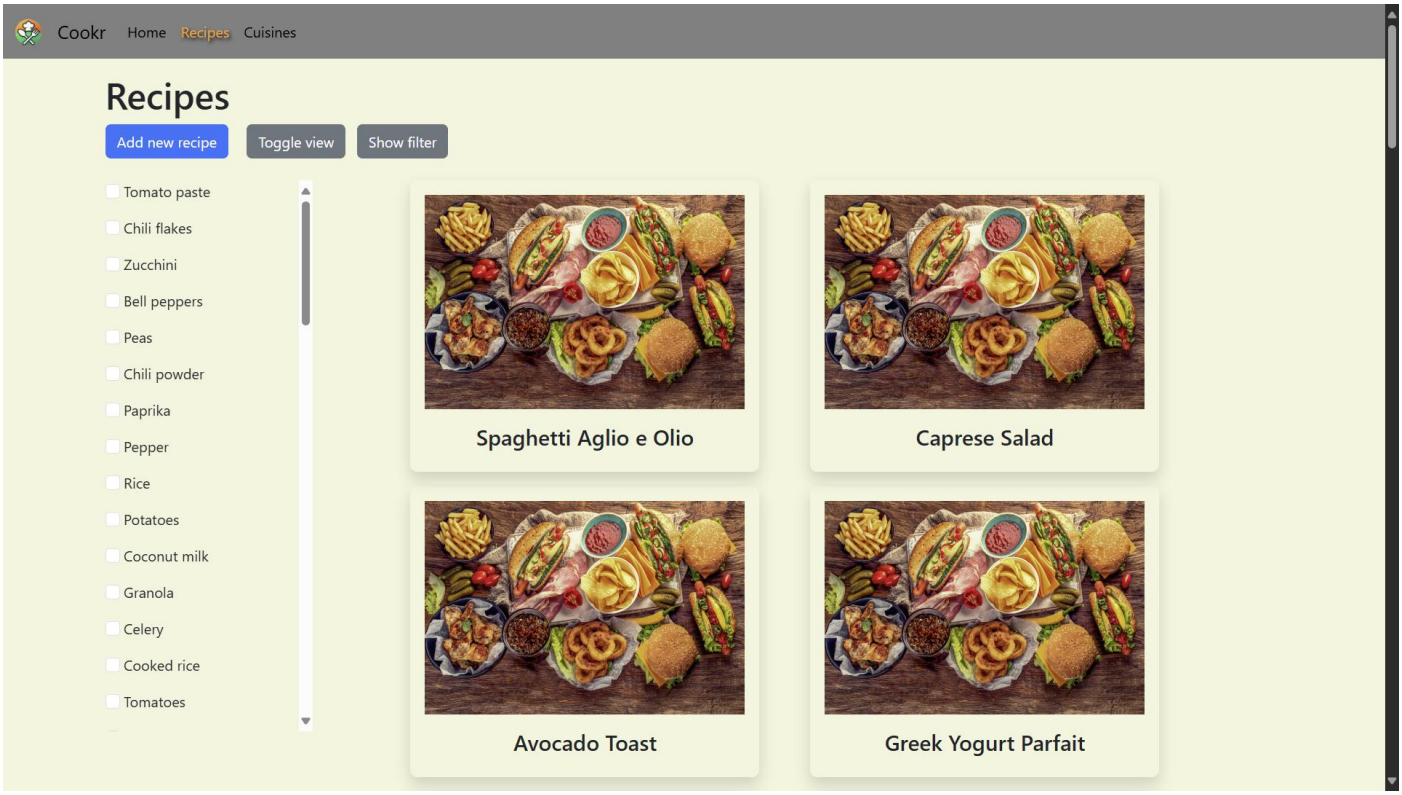


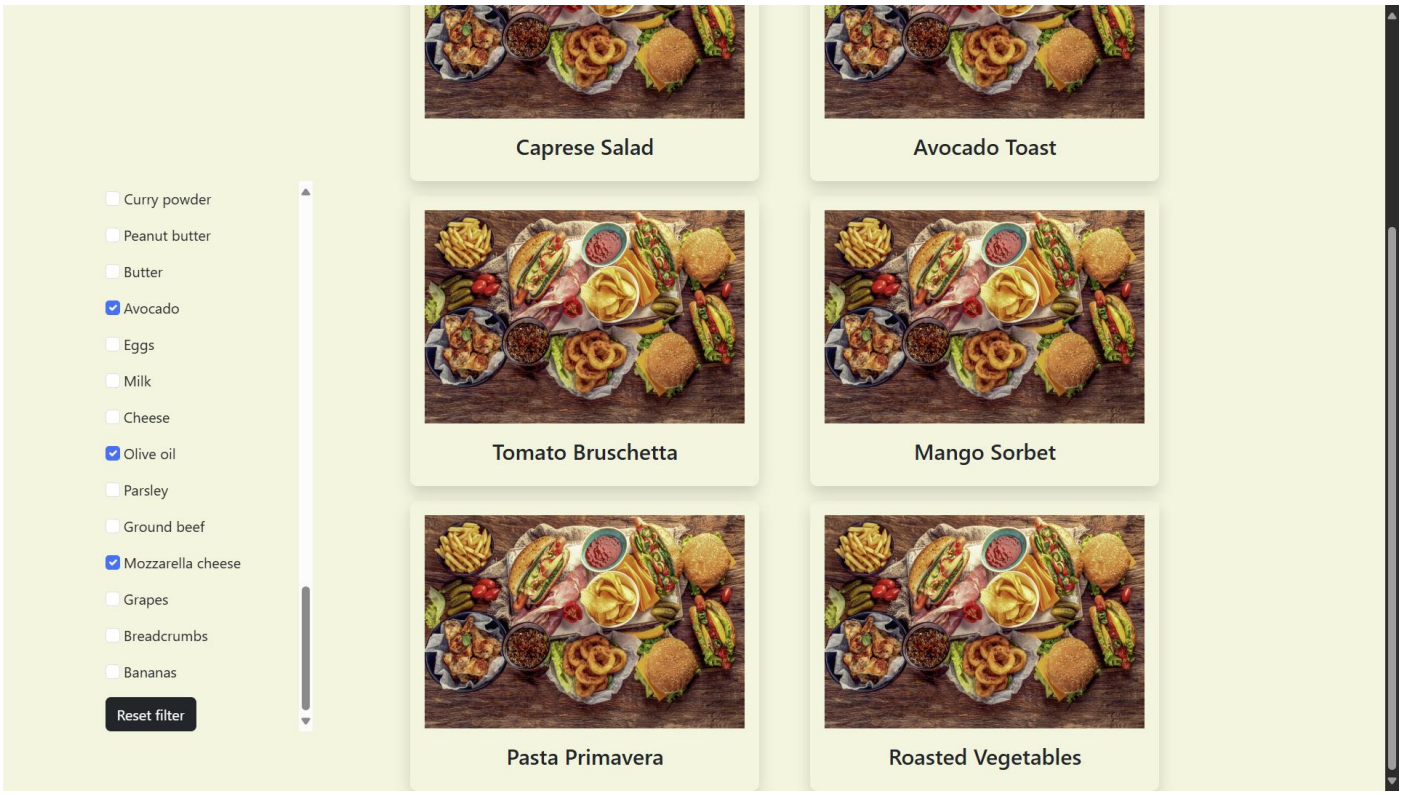
Es gibt zwei Ansichten für die Rezepte: eine detailliertere und eine kompakte Darstellung. Bei der detaillierten werden alle Informationen über das Rezept dargestellt (Bild, Name, Beschreibung, Zutaten). Dazu wird die *RecipeCard.svelte* Komponente verwendet. Bei der kompakten Darstellung reduzieren sich die Informationen auf das Bild und den Namen. Bei dieser Ansicht wird die *RecipeCardCompact* Komponente angezeigt. Standardmässig wird die detaillierte Ansicht angezeigt. Zwischen den beiden Ansichten kann durch einen Klick auf den Button *Toggle view* gewechselt werden.

Bei beiden Ansichten kann die Detailseite (siehe weiter unten) eines Rezeptes durch den Klick auf den Namen des Rezeptes aufgerufen werden.




Curry powder
Peanut butter
Butter
Avocado
Eggs
Milk
Cheese
Olive oil
Parsley
Ground beef
Mozzarella cheese
Grapes
Breadcrumbs
Bananas
Reset filter
Avocado Toast
A simple and healthy avocado spread on toast.
Avocado
Bread
Tomato Bruschetta


Cookr
Home
Recipes
Cuisines
Recipes
Add new recipe
Toggle view
Show filter
Tomato paste
Chili flakes
Zucchini
Bell peppers
Peas
Chili powder
Paprika
Pepper
Rice
Potatoes
Coconut milk
Granola
Celery
Cooked rice
Tomatoes
Spaghetti Aglio e Olio
Caprese Salad
Avocado Toast
Greek Yogurt Parfait



Neben dem Button, um die Ansicht zu wechseln gibt es noch einen weiteren Button *Show filter*. Wenn dieser betätigt wird, erscheint auf der linken Seite des Bildschirmes eine Auflistung von Zutaten und jeweils einer Checkbox. Hier kann der Benutzer nun Zutaten auswählen, die er zum Beispiel gerade zuhause hat oder mit denen er kochen möchte. Bei der Betätigung einer Checkbox werden die Rezepte gefiltert, sodass nur noch diejenigen angezeigt werden, die Zutaten enthalten, die der Benutzer selektiert hat. Mit dem Button ganz unten *Reset filter* kann der Filter zurückgesetzt werden, damit wieder alle Rezepte angezeigt werden.

Die Ansicht ist responsive und zeigt die Elemente einerseits passen zur Fenstergrösse an und auch je nachdem welche Ansicht aktiv ist sowie ob die Filter sichtbar sind oder nicht.

Ein weiterer Button *Ingredients chart* führt den Benutzer zur Anzeige eines Diagramms für Zutaten (siehe weiter unten).

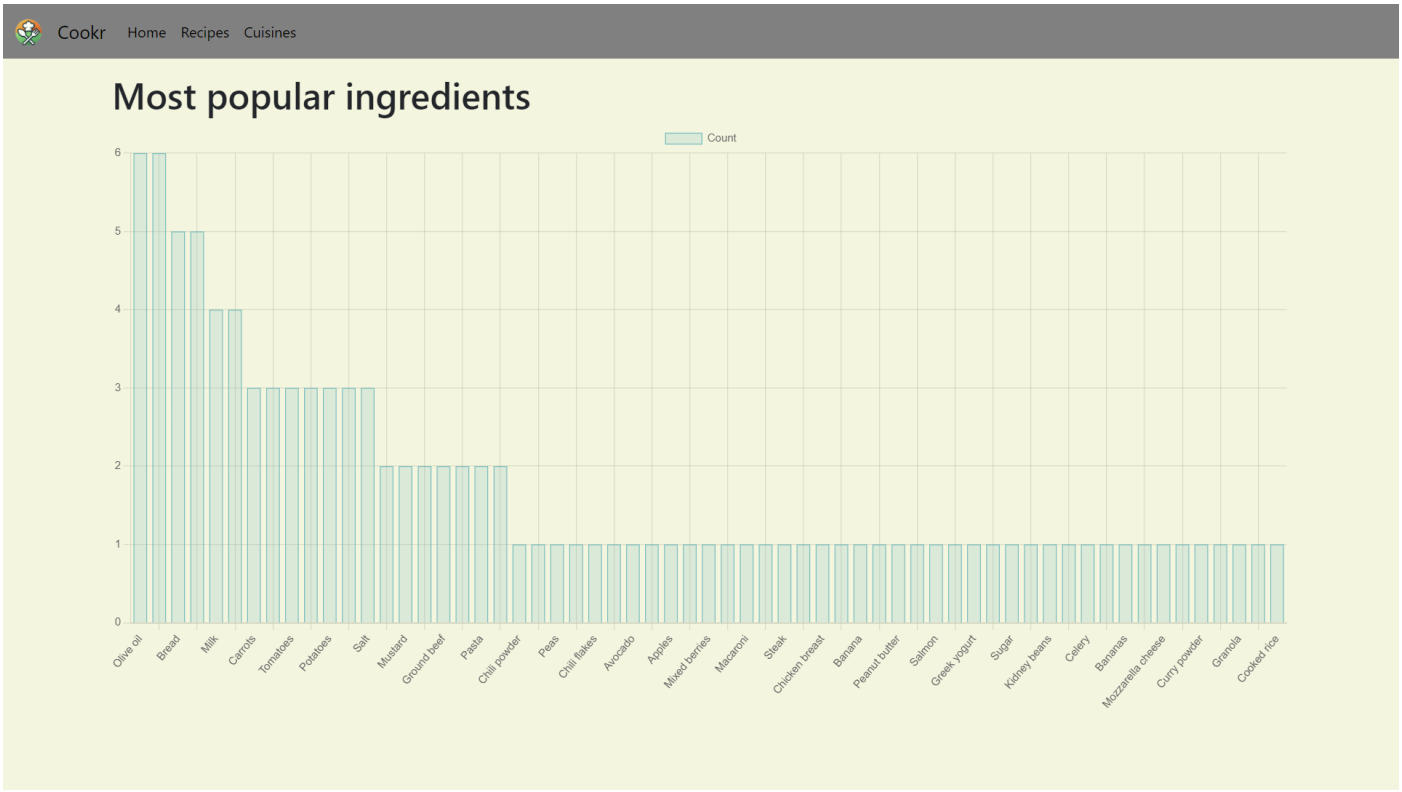
Als letzten Button auf der Seite gibt es noch *Add new recipe*. Damit gelangt der Benutzer zur Seite zum Hinzufügen eines neuen Rezeptes (siehe weiter unten).

Dateien

- routes/recipes/+page.svelte
- routes/recipes/+page.server.js
- lib/components/RecipeCard.svelte
- lib/components/RecipeCardCompact.svelte

Chart für Zutaten

Route: /ingredientsChart



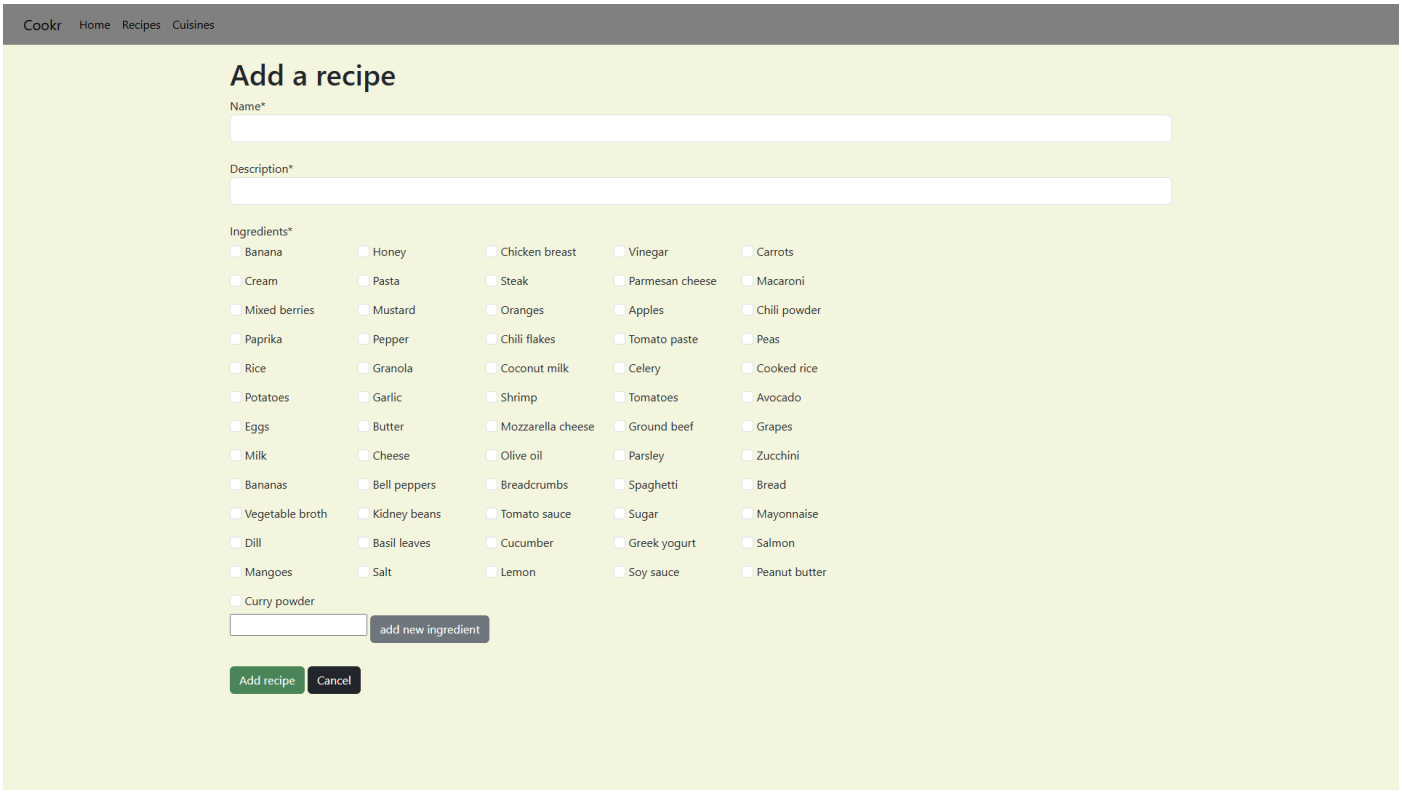
Bei einem Klick auf den Button *Ingredients chart* auf der Übersichtsseite oder durch die Eingabe der URL wird der Benutzer auf eine Seite geleitet, die ein Balkendiagramm mit Zutaten anzeigt. Es ist ersichtlich, in wie vielen Rezepten eine einzelne Zutat enthalten ist. So kann der Benutzer sehen, welche Zutaten wie wichtig sind beim Kochen.

Dateien

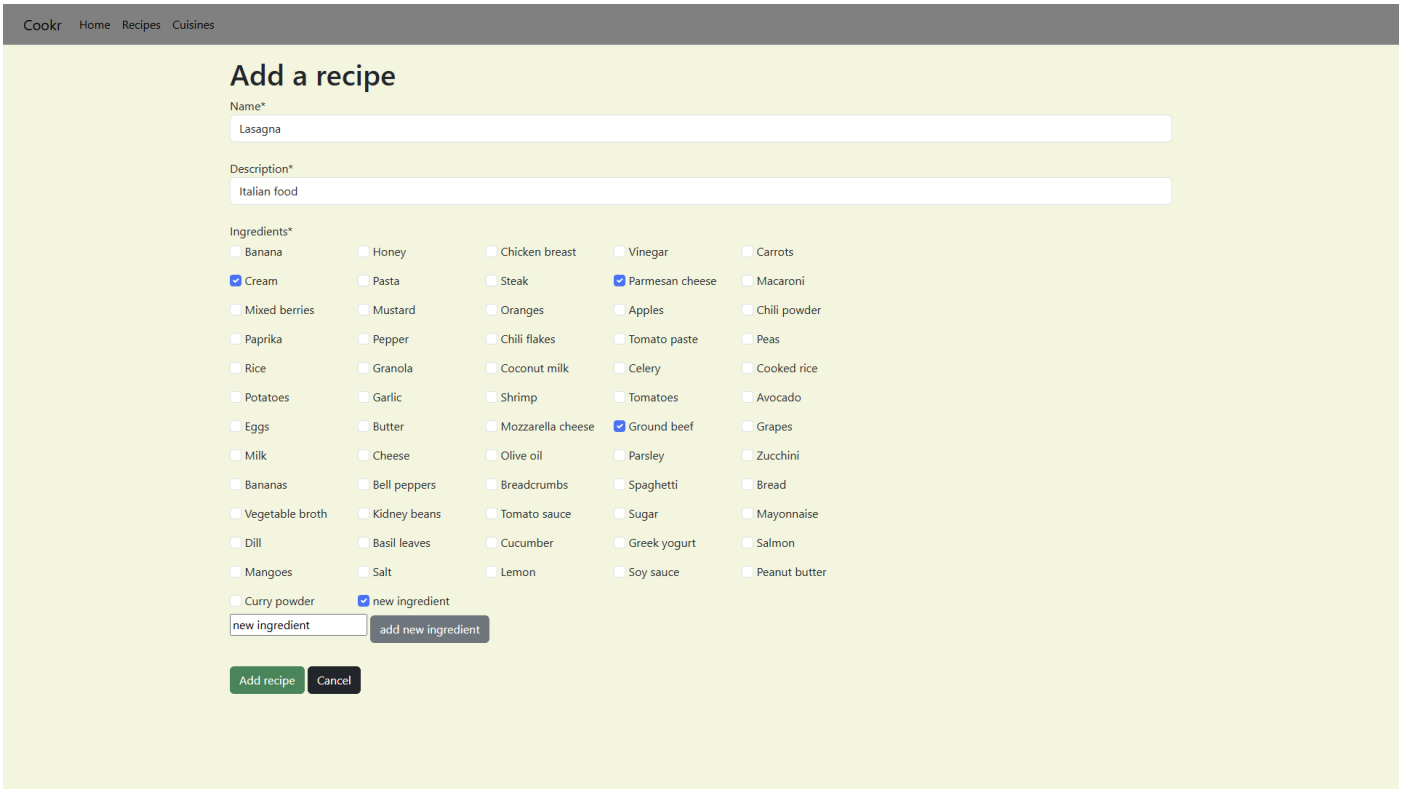
- routes/ingredientsChart/+page.svelte
- routes/ ingredientsChart/page.server.js

Rezept hinzufügen

Route: /recipes/create



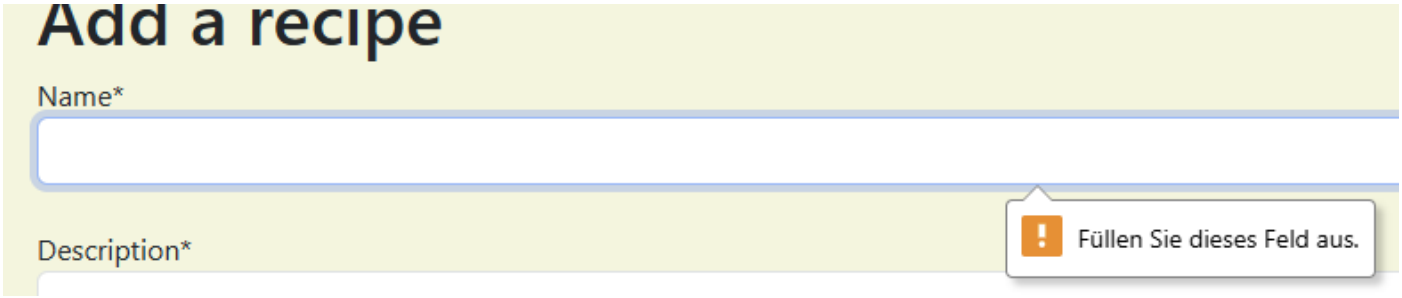
Diese Seite ermöglicht es dem Benutzer neue Rezepte zu erstellen und in der Datenbank zu speichern.

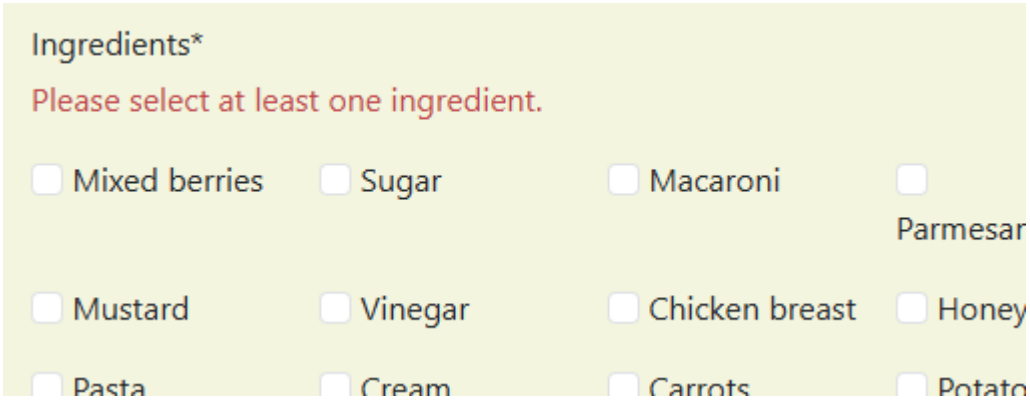


Die Eingabeelemente sind der Name, die Beschreibung und die Zutaten. Die Auswahlmöglichkeiten für Zutaten besteht aus allen Zutaten, die bei bestehenden Rezepten enthalten sind. Der Benutzer kann diese mit einer Checkbox selektieren. Wenn der Benutzer eine Zutat auswählen möchte, die es bisher noch nicht gibt, so kann er

den Namen der Zutat in einem Textfeld eingeben und den Button *add new ingredient* tätigen. Daraufhin erscheint die neue Zutat neben den bestehenden und ist direkt selektiert.

Mit dem Button *Add recipe* kann der Benutzer das eingegebene Rezept speichern lassen. Dabei wird das Formular validiert, um sicherzustellen, dass der Name und die Beschreibung nicht leer sind, sowie mindestens eine Zutat ausgewählt wurde. Falls dem nicht so ist, wird es dem Benutzer mit einer Meldung mitgeteilt. Nur wenn alle Bedingungen erfüllt sind, wird das Rezept gespeichert. Bei erfolgreicher Speicherung wird dem Benutzer die Detailseite des soeben erstellten Rezeptes angezeigt.





Dateien

- routes/recipes/create/+page.svelte
- routes/recipes/create/+page.server.js

Detailseite Rezept

Route: /recipes/[recipe_id]

Bei einem Klick auf ein Rezept auf der Übersichtsseite oder durch die Eingabe der URL wird die Detailseite eines Rezeptes angezeigt. In der Mitte des Bildschirms sind alle Informationen eines Rezeptes ersichtlich: der Name, ein Bild, die Beschreibung des Gerichts und die Zutaten.

Durch den Button *Back* gelangt man zurück zur Übersicht.

Mit dem Button *Delete* kann das Rezept gelöscht werden, woraufhin der Benutzer ebenfalls zurück zur Übersicht geleitet wird.

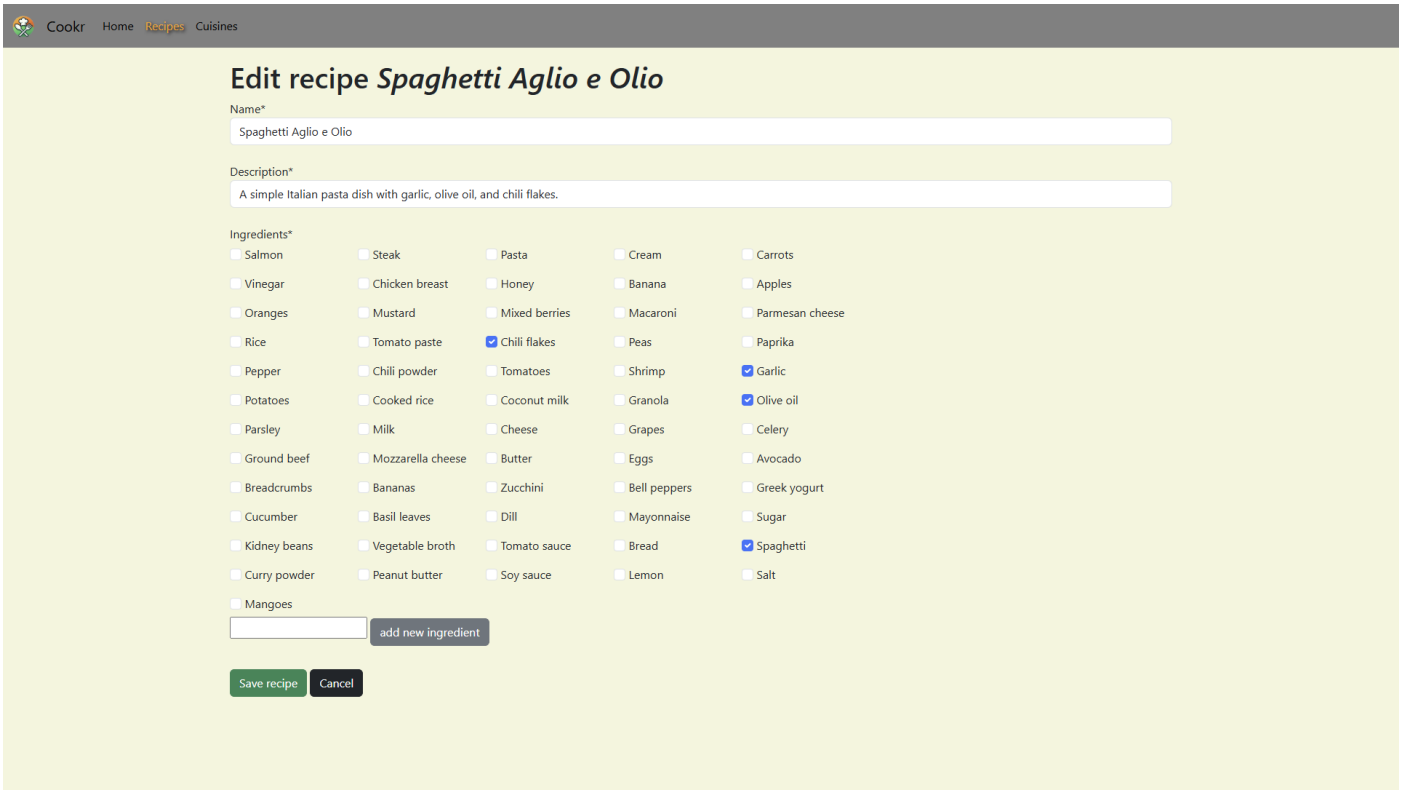
Der Button *Edit* leitet den Benutzer weiter auf eine Seite, auf der er das Rezept bearbeiten kann (siehe weiter unten).

Dateien

- routes/recipes/[recipe_id]/+page.svelte
- routes/recipes/[recipe_id]/+page.server.js

Rezept bearbeiten

Route: /recipes/[recipe_id]/update



Über die Detailseite eines Rezeptes kann die Page zum Bearbeiten des Rezeptes aufgerufen werden. Dabei gibt es die gleichen Eingabeelemente wie beim Erstellen eines neuen Rezeptes. Die Werte Name und Beschreibung werden bereits in die Felder geschrieben, sodass sie nur noch angepasst werden können. Bei den Zutaten sind diejenigen bereits aktiviert, die im Rezept hinterlegt sind. Zum Speichern der Änderungen kann der Benutzer den Button *Save Recipe* klicken, woraufhin er wiederrum auf die Detailseite geleitet wird. Wie beim Erstellen eines neuen Rezeptes erfolgt auch hier eine Validierung des Formulars.

Dateien

- routes/recipes/[recipe_id]/update/+page.svelte
- routes/recipes/[recipe_id]/update/+page.server.js

Lieblingsküchen

Route: /cuisines

Die Seite *Cuisines* zeigt alle in MongoDB gespeicherten Entitäten sortiert nach deren Attribut *id.* Von jedem Element werden der Name und der Rang zum Zeitpunkt, als es aus der Datenbank geladen wurde, angezeigt.

Der Benutzer hat die Möglichkeit, die Elemente per Drag and Drop neu anzuordnen. Die Nummerierung bleibt bestehen, damit der Nutzer sieht, wie die ursprüngliche Reihenfolge war. Sobald der Benutzer zufrieden ist mit der Anordnung, kann er den Button *Save order* betätigen. Daraufhin werden die Attribute *id* auf den Entitäten mit der neuen Nummerierung gemäss der Anordnung des Benutzers überschrieben. Die Seite und damit die Daten werden neu geladen und somit auch die aktualisierte Reihenfolge angezeigt.

Dateien

- routes/cuisines/+page.svelte
- routes/cuisines/+page.server.js

Erweiterungen

Rezepte nach Zutaten filtern

Auf der Seite /recipes kann ein Filter eingeblendet werden, mit dem man die Rezepte nach Zutaten filtern kann. Wenn eine Zutat aus der Liste selektiert oder deaktiviert wird, wird immer eine Funktion aufgerufen, der den Wert *checked* auf den Zutaten entweder auf *true* oder *false* setzt. Danach wird die Liste an sichtbaren Rezepten aktualisiert, damit nur die erwünschten Rezepte angezeigt werden.

Dateien

- routes/recipes/+page.svelte

Formular mit Checkboxen zum Auswählen von Zutaten

Auf den Seiten /recipes/create und /recipes/[recipe_id]/update kann der Benutzer jede Zutat mit einer Checkbox selektieren. Beim Speichern eines Rezeptes werden alle selektierten Zutaten aus dem Formular genommen und in das Array vom Attribut vom Rezept geschrieben. Dadurch werden alle selektierten Zutaten auf das Rezept gespeichert.

Dateien

- routes/recipes/create/+page.svelte
- routes/recipes/create/+page.server.js
- routes/recipes/[recipe_id]/update/+page.svelte
- routes/recipes/[recipe_id]/update/+page.server.js

Formular zum Hinzufügen neuer Zutaten

Auf den Seiten /recipes/create und /recipes/[recipe_id]/update kann der Benutzer neben den existierenden Zutaten auch neue erstellen und auswählen. Dazu gibt es ein Eingabefeld für den Namen der neuen Zutat und einen Button zum Hinzufügen. Beim Klick auf den Button wird eine Funktion aufgerufen, der den Wert aus dem Eingabefeld liest und ihn als Objekt dem Array der Zutaten hinzufügt. Die neu erstellte Zutat wird nach allen anderen angezeigt und weil das Attribut *checked* auf *true* gesetzt wurde, ist die Checkbox direkt selektiert.

Dateien

- routes/recipes/create/+page.svelte
- routes/recipes/create/+page.server.js
- routes/recipes/[recipe_id]/update/+page.svelte
- routes/recipes/[recipe_id]/update/+page.server.js

Formular Validierung

Auf den Seiten /recipes/create und /recipes/[recipe_id]/update wird das Formular validiert bevor es durchgeführt wird. Zunächst sind die beiden Eingabefelder für den Namen und die Beschreibung mit dem Attribut *required* versehen, sodass diese einen Wert haben müssen, damit das Rezept gespeichert werden kann. Des Weiteren wird mit der Funktion, die ausgeführt wird bei *onsubmit* überprüft, ob mindestens eine Zutat selektiert ist. Falls dies nicht der Fall ist, wird das Formular nicht abgeschickt und der Hinweis wird angezeigt.

Dateien

- routes/recipes/create/+page.svelte
- routes/recipes/create/+page.server.js
- routes/recipes/[recipe_id]/update/+page.svelte
- routes/recipes/[recipe_id]/update/+page.server.js

MongoDB Aggregation zum Laden der Zutaten

Auf den Seiten /recipes, /recipes/create und /recipes/[recipe_id]/update werden alle in der Datenbank vorhandenen Rezepte geladen. Da diese in MongoDB jeweils in einem Array unter den Rezepten gespeichert sind und nicht in

einer separaten Collection, müssen sie mittels einer Aggregation Pipeline geladen werden. Dazu wurde in *db.js* die passende Pipeline erstellt und durchgeführt, wenn die Methode aufgerufen wird.

Dateien

- lib/db.js

Drag-and-Drop bei Lieblingsküchen

Auf der Seite /cuisines kann der Benutzer die Elemente in der Rangliste per Drag-and-Drop verschieben und anordnen. Dies wurde mit einer Library namens «svelte-dnd-action» umgesetzt. Die Elemente werden in einem Array gespeichert und wenn es auf der *section*, in der die Elemente sind, ein Drag-and-Drop Event gab, wird der Array sofort neu sortiert. Dieser Array (mit der neuen Reihenfolge) wird dann auch zum Speichern an die Datenbank übergeben.

Dateien

- routes/cuisines/+page.svelte
- routes/cuisines/+page.server.js

Animation auf Drag-and-Drop Elementen

Auf der Seite /cuisines, wenn der Benutzer ein Element per Drag-and-Drop neu platziert hat, verschieben sich die Elemente nicht sofort, sondern mit einer Animation.

Dateien

- routes/cuisines/+page.svelte

Chart für Zutaten

Auf der Seite /ingredientsChart wird ein Balkendiagramm angezeigt, welche mit der Library *chart.js erstellt* wurde. Dabei stellt die y-Achse die Anzahl an Rezepten, in denen eine Zutat enthalten ist, dar.

Dateien

- routes/ingredientsChart/+page.svelte
- routes/ingredientsChart/+page.server.js

Error-Handling in DB-Funktionen

Da bei einer Schnittstelle zu einer Datenbank Fehler auftreten können und der Benutzer dabei nur einen Fehlercode von 500 sehen würde, wurde bei den Funktionen in *db.js* ein Error-Handling eingebaut. Bei Exceptions in try-catch Blöcken und anderen Fällen (bsp. Keine Entität mit gegebener Id gefunden) wird der Fehler einerseits auf der Konsole ausgegeben und andererseits als Rückgabewert dem Aufrufer zurückgegeben. Bei den einzelnen Pages wird dann überprüft, ob die Antwort von der Datenbank erfolgreich war. Falls nicht, wird dem Benutzer die Fehlernachricht angezeigt.

Dieser Screenshot ist ein Beispiel, bei dem über die URL eine Detailseite eines Rezeptes aufgerufen werden wollte, welches nicht in der Datenbank vorhanden ist. Die Nachricht wird von der Funktion in *db.js* retourniert und in *+page.server.js* und *+page.svelte* gehandhabt.

Dateien

- lib/db.js
- routes/recipes/+page.svelte
- routes/ recipes/+page.server.js
- routes/recipes/create/+page.svelte
- routes/recipes/create/+page.server.js
- routes/recipes/[recipe_id]/+page.svelte
- routes/recipes/[recipe_id]/+page.server.js
- routes/recipes/[recipe_id]/update/+page.svelte
- routes/recipes/[recipe_id]/update/+page.server.js
- routes/cuisines/+page.svelte
- routes/cuisines/+page.server.js

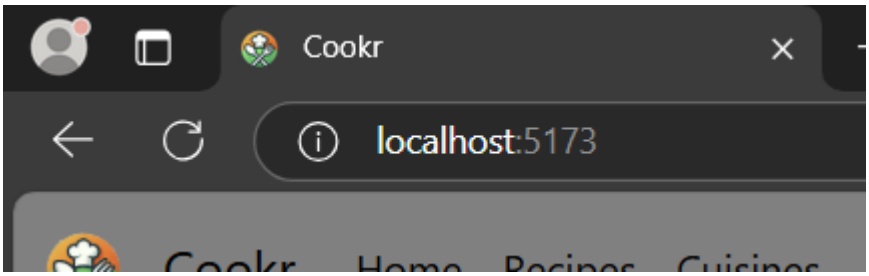
Weiterleitung nach Formular

Beim Speichern eines neuen Rezeptes oder eines angepassten Rezeptes wird der Benutzer zur Detailseite weitergeleitet, sofern die Aktion erfolgreich war. Ebenfalls wird nach der Löschung eines Rezeptes auf der Detailseite wieder die Übersichtsseite aller Rezepte angezeigt, wenn die Löschung erfolgreich war.

- routes/recipes/create/+page.server.js
- routes/recipes/[recipe_id]/+page.server.js
- routes/recipes/[recipe_id]/update/+page.server.js

Favicon

Im Browser wird beim Tab der Website das Logo als Favicon angezeigt. Standardmässig ist ein svelte-Logo dort sichtbar. Dieses wurde mit einem von ChatGPT generierten Logo für die Website ausgetauscht.



Dateien

- app.html