



Shopping Cart

You're an owner of a newly-started e-commerce store, and you want to create a basic web service to cater to your customers. Since you're just starting up, you want to keep it simple, with the following schema:

User -> Add Items to Cart -> Order with Items in the Cart -> Done!

You're planning to use `gorm` (<https://github.com/jinzhu/gorm/>) as the ORM, `gin` (<https://github.com/gin-gonic/gin/>) as the web framework and `ginkgo` (<https://github.com/onsi/ginkgo>) for any tests you want to add.

You're keeping the following flow in mind for the user:

1. When a user will come to your platform and signup (POST `/users` API is called), a new User account gets created.
2. The user needs to be logged in order to create a cart. If the user already has an account, the user will login using POST `/users/login` API which will return a token for further requests in the user's session. A user can only be logged in from a single device at a time i.e. there'll always be a single token for the user.
3. When the user starts shopping and selects Items, the Items will get added to the Cart. This will be done by POST `/carts` API. A single user can have only a single cart, so you'll need to identify the Cart by the User's ID.

Note: For sake of simplicity let's assume that we don't want to manage inventory yet i.e. we don't have to keep track of the number of items, whether it's in stock or out of stock, etc. 4. The cart will get converted into an order when the POST `/orders` API is called. 5. There should also be listing endpoints for User, Items, Carts and Orders.

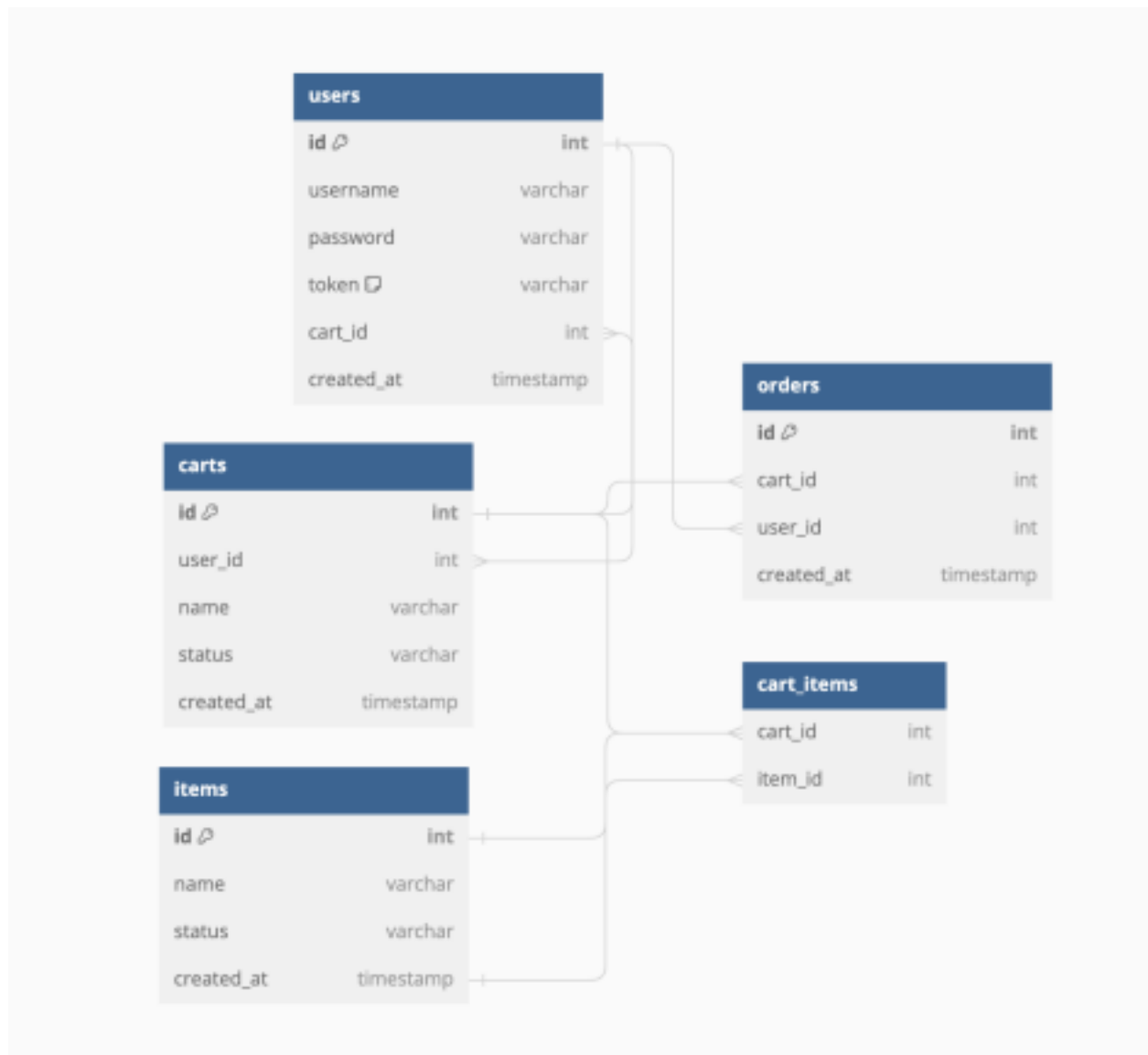
Endpoints Summary as below:

Endpoint URL	Description
POST /users	Creates a new User
GET /users	List all users
POST /users/login	Login for existing user based on username and password
POST /items	Creates an Item
GET /items	List all items
*POST /carts/ GET /carts	Create and adds Items to the cart List all carts
*POST /orders	Pass the cart id to convert it to an order
GET /orders	List all orders

*The user's token must be present in the cart related endpoint request to identify which user the cart belongs to.

A B C D E

A general overview of the Entities that need to be used:



Now that you've the backend ready, time to show it in the UI!

Create a React web app with the following screens in order:

1. Create a basic User Login screen where the user can enter the username and password to List the Items in the shopping portal. On login failure show a `window.alert()` saying Invalid username/password.
2. On successful login take the user to the List Items screen. The List Items screen is where the user can see all the items. Clicking on an item in the items list should add that item to the cart.
3. Show a Checkout button at the top of this screen which the user will use to place an order. Also next to the Checkout button show 2 buttons:
 - a. Cart button to list all the added Items in the cart. Clicking on this button should show all the cart items (i.e. `cart_id`, `item_id`) in a toast or `window.alert()`.
 - b. Order History button to list all the placed orders for the user. Clicking on this button should show all the placed Order ids in a toast or `window.alert()`.
4. On clicking the checkout button, the cart should get converted to an order. You can show the List Items screen again with a toast saying Order successful.

A B C D E₄