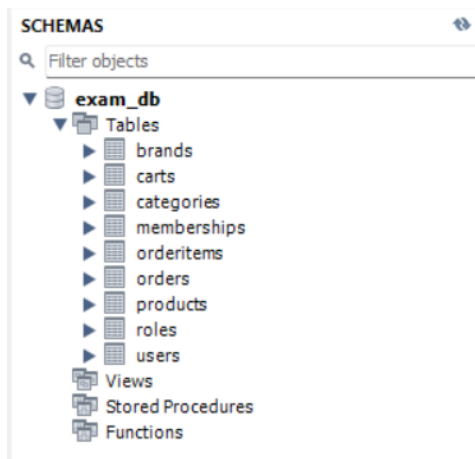


Reflection Report

Database

All the tables



```
1 • SELECT * FROM exam_db.brands;
```

id	name	createdAt	updatedAt
1	Apple	2025-06-07 18:06:38	2025-06-07 18:06:38
2	Samsung	2025-06-07 18:06:38	2025-06-07 18:06:38
3	Xiaomi	2025-06-07 18:06:38	2025-06-07 18:06:38
4	MXQ	2025-06-07 18:06:38	2025-06-07 18:06:38
5	TEST_BRAND2	2025-06-07 18:06:38	2025-06-07 18:06:38
NULL	NULL	NULL	NULL

```
1 • SELECT * FROM exam_db.carts;
```

id	quantity	createdAt	updatedAt	userId	productId
NULL	NULL	NULL	NULL	NULL	NULL

```
1 • SELECT * FROM exam_db.categories;
```

id	name	createdAt	updatedAt
1	Phones	2025-06-07 18:06:38	2025-06-07 18:06:38
2	TVs	2025-06-07 18:06:38	2025-06-07 18:06:38
3	Watches	2025-06-07 18:06:38	2025-06-07 18:06:38
4	Desktops	2025-06-07 18:06:38	2025-06-07 18:06:38
5	Laptops	2025-06-07 18:06:38	2025-06-07 18:06:38
6	Tablets	2025-06-07 18:06:38	2025-06-07 18:06:38
7	TEST_CATEGORY2	2025-06-07 18:06:38	2025-06-07 18:06:38
NULL	NULL	NULL	NULL

```
1 • SELECT * FROM exam_db.memberships;
```

Result Grid							
Filter Rows:							
Edit:							
Export/Import:							
	id	name	min	max	discount	createdAt	updatedAt
▶	1	Bronze	0	15	0	2025-06-07 18:06:37	2025-06-07 18:06:37
	2	Silver	15	30	15	2025-06-07 18:06:37	2025-06-07 18:06:37
	3	Gold	30	9999	30	2025-06-07 18:06:37	2025-06-07 18:06:37
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
1 • SELECT * FROM exam_db.orders;
```

Result Grid							
Filter Rows:							
Edit:							
Export/Import:							
	id	status	totalQuantity	discountApplied	orderNumber	createdAt	updatedAt
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
1 • SELECT * FROM exam_db.products;
```

Result Grid											
Filter Rows:											
Edit:											
Export/Import:											
Wrap Cell Content: 12											
	id	title	description	price	quantity	image	isDele	createdAt	updatedAt	CategoryId	BrandId
▶	1	iPhone 6s Plus ...	3D Touch. 12MP photos. 4K video.	649.00	2	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	1	1
	2	Apple TV 2016	The future of television is here.	599.00	10	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	2	1
	3	iPhone SE 32/6...	A big step for small	499.00	15	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	1	1
	4	Apple Watch S...	You. At a glance	399.00	1	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	3	1
	5	21.5-inch iMac ...	Retina. Now in colossal and ginor...	599.00	20	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	4	1
	6	MacBook Pro wi...	Stunning Retina Display	99.00	18	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	5	1
	7	9.7-inch iPad P...	3D Touch. 12MP photos. 4K video.	99.00	18	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	6	1
	8	Mac Pro Quad-...	Better for creativity on an epic scale	2999.00	27	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	4	1
	9	iPad Air 2 32/6...	Light. Heavyweight	399.00	55	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	6	1
	10	Samsung Galax...	Dual Sim card - Lavender color	398.00	15	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	1	2
	11	Samsung Galax...	Dual Sim card - Black color	398.00	15	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	1	2
	12	Xiaomi 4K Ultra ...	360 Bluetooth & IR remote control...	49.00	38	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	2	3
	13	Android 10.1 M...	Bluetooth & IR remote control	49.00	14	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	2	4
	14	Samsung Galax...	Android table - perfect for kids	20.00	8	http://images.restapi.co.z...	0	2025-06-07 ...	2025-06-07 18:06:38	6	2
	15	TEST_PRODUCT	Test desc	42.42	5	NULL	1	2025-06-07 ...	2025-06-07 18:06:38	7	5
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
1 • SELECT * FROM exam_db.roles;
```

Result Grid				
Filter Rows:				
Edit:				
	id	name	createdAt	updatedAt
▶	1	Admin	2025-06-07 18:06:37	2025-06-07 18:06:37
	2	User	2025-06-07 18:06:37	2025-06-07 18:06:37
*	NULL	NULL	NULL	NULL

1 • `SELECT * FROM exam_db.users;`

id	firstname	lastname	username	email	password	address	phone	role_id	membership_id	total_items_purchased	createdAt	updatedAt
1	Admin	Support	Admin	admin@noroff.no	\$2b\$10\$UI/xMoU8Dmixf...	Online	911	1	1	0	2025-06-07 18:06:37	2025-06-07 18:06:37
2	Nils	Nilsen	Nils	nils@hotmail.com	\$2b\$10\$ftny4moA/VgaU...	Oslo	123	2	1	0	2025-06-08 08:14:04	2025-06-08 08:14:04
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

The relationship between the tables

- **Role → User (1→N):** Role.id (PK) → User.role_id (FK). One role (e.g. Admin) can be assigned to many users, but each user has exactly one role.
- **Membership → User (1→N):** Membership.id (PK) → User.membership_id (FK). One membership level (Bronze/Silver/Gold) applies to many users; each user holds exactly one level at a time.
- **User → Cart (1→N):** User.id (PK) → Cart.user_id (FK). One user can have multiple cart entries (one per product), but each cart entry belongs to a single user.
- **Cart → Product (1→N):** Product.id (PK) → Cart.product_id (FK). One product can appear in many users' carts; each cart entry references one product.
- **Category → Product (1→N):** Category.id (PK) → Product.category_id (FK). Each category groups many products; each product belongs to one category.
- **Brand → Product (1→N):** Brand.id (PK) → Product.brand_id (FK). Each brand can have many products; each product has one brand.
- **User → Order (1→N):** User.id (PK) → Order.user_id (FK). One user can place many orders; each order belongs to one user.
- **Order → OrderItem (1→N):** Order.id (PK) → OrderItem.order_id (FK). One order can include multiple order items; each order item is linked to a single order.
- **Product → OrderItem (1→N):** Product.id (PK) → OrderItem.product_id (FK). One product can be part of many order items; each order item refers to one product.

Jira

<div> <div> <div></div> <div>EC Sprint 1</div> <div>12 May – 19 May (4 work items)</div> </div> <div> <div>0</div> <div>0</div> <div>0</div> </div> <div>Start sprint</div> <div>...</div> </div>
<div> <div> <div></div> <div>EC-6</div> <div>Model all tables (Users, Roles, Memberships, Products, Categories, Brands, Carts, Orders, OrderItems) in 3NF.</div> </div> <div>DATABASE & SEED DATA</div> <div>TO DO</div> <div> <div></div> <div></div> </div> </div>
<div> <div> <div></div> <div>EC-7</div> <div>Write Sequelize models & migrations Define each table, fields, data types, timestamps.</div> </div> <div>DATABASE & SEED DATA</div> <div>TO DO</div> <div> <div></div> <div></div> </div> </div>
<div> <div> <div></div> <div>EC-8</div> <div>Implement /init endpoint Fetch Noroff API data, seed Products, Categories, Brands. Seed Roles (Admin/User)...</div> </div> <div>DATABASE & SEED DATA</div> <div>TO DO</div> <div> <div></div> <div></div> </div> </div>
<div> <div> <div></div> <div>EC-9</div> <div>Write init-endpoint error handling Idempotency check (run only once) and clear error messages.</div> </div> <div>DATABASE & SEED DATA</div> <div>TO DO</div> <div> <div></div> <div></div> </div> </div>

+

Create

☐

EC Sprint 2

19 May – 26 May (5 work items)

0

0

0

Start sprint

...

EC-10

Build POST /auth/register Validate input, hash password, assign default User role and Bronze membership.

AUTHETICATION & AU...

TO DO

-

EC-11

Build POST /auth/login Verify username/email + password, issue JWT (2 h expiry).

AUTHETICATION & AU...

TO DO

-

EC-12

JWT middleware Extract/verify token, attach req.user.

AUTHETICATION & AU...

TO DO

-

EC-13

Role-check middleware Block non-admin users from admin routes (return 403).

AUTHETICATION & AU...

TO DO

-

EC-14

Error-handling & validation Return proper HTTP status codes & JSON error structure on bad creds or missi...

AUTHETICATION & AU...

TO DO

-

EC Sprint 3

26 May – 2 Jun (5 work items)

0

0

0

Start sprint

...

EC-15

CRUD endpoints for Products/Categories/Brands GET, POST, PUT, DELETE (soft-delete for Products).

PRODUCT CATALOG & ...

TO DO

-

EC-16

Raw-SQL /search endpoint Search by partial name, category name, or brand name (parameterized LIKE join...

PRODUCT CATALOG & ...

TO DO

-

EC-17

Cart endpoints POST /cart (add item), PUT /cart/id (update qty), GET /cart. Stock check before adding.

PRODUCT CATALOG & ...

TO DO

-

EC-18

Checkout logic POST /cart/checkout/now: create Order + OrderItems, copy price_at_order, apply current m...

PRODUCT CATALOG & ...

TO DO

-

EC-19

Membership discount recalculation After checkout, update user's total items → new membership level if th...

PRODUCT CATALOG & ...

TO DO

-

EC Sprint 4

2 Jun – 9 Jun (6 work items)

0

0

0

Start sprint

...

<div>EC-20</div>	Admin front-end scaffolding Express + EJS + Bootstrap layout, navbar, footer partials.	<div>ADMIN UI & DOCS & ...</div>	<div>TO DO ▾</div>	<div>-</div>	<div></div>
<div>EC-21</div>	CRUD views Pages/forms for Users, Products, Categories, Brands, Memberships, Orders (status update).	<div>ADMIN UI & DOCS & ...</div>	<div>TO DO ▾</div>	<div>-</div>	<div></div>
<div>EC-22</div>	Integrate API calls Client-side fetch or Axios to back-end endpoints; handle auth token in headers.	<div>ADMIN UI & DOCS & ...</div>	<div>TO DO ▾</div>	<div>-</div>	<div></div>
<div>EC-23</div>	Swagger UI at /doc Add and document all endpoints with example payloads & responses.	<div>ADMIN UI & DOCS & ...</div>	<div>TO DO ▾</div>	<div>-</div>	<div></div>
<div>EC-24</div>	Automated tests Jest + Supertest for each CRUD endpoint, auth flows, search, cart/checkout.	<div>ADMIN UI & DOCS & ...</div>	<div>TO DO ▾</div>	<div>-</div>	<div></div>
<div>EC-25</div>	CORS & config polishing Back-end CORS whitelist, README update, environment variable instructions.	<div>ADMIN UI & DOCS & ...</div>	<div>TO DO ▾</div>	<div>-</div>	<div></div>

	y		June	
Sprints		EC S...	EC S...	EC Sprint 4
<input type="checkbox"/> > <u>EC-1 Database & Seed Data</u>				
<input type="checkbox"/> > <u>EC-2 Authetication & Authorization</u>				
<input type="checkbox"/> > <u>EC-4 Product Catalog & Checkout</u>				
<input type="checkbox"/> > <u>EC-5 Admin UI & DOCS & Testing</u>				

Mon	Tue	Wed	Thu	Fri
2	3	4	5	6
EC Sprint 3				
EC Sprint 4				
9	10	11	12	13
EC Sprint 4				

Mon	Tue	Wed	Thu	Fri
12	13	14	15	16
🔗 EC Sprint 1				
19	20	21	22	23
🔗 EC Sprint 1				
🔗 EC Sprint 2				
26	27	28	29	30
🔗 EC Sprint 2				
🔗 EC Sprint 3				

.env-fil

```

aug24ft-ep-ca-1-JonBergeroeRyssdal > back-end >
1 DB_NAME=exam_db
2 DB_USER=root
3 DB_PASS=XXXXXXXXXXXXXXXXXXXX
4 DB_HOST=localhost
5 DB_PORT=3306
6 JWT_SECRET=XXXXXXXXXXXXXXXXXXXX
7
8
9

```

1. Discussion of the progression of the project

Balancing a full-time job and family commitments made it challenging to carve out consistent blocks of time for the project. On weeknights I managed to code one to three evenings—usually one or two hours each session—depending on how tired I was. Every weekend, however, I settled in for longer stretches, typically four to eight hours on both Saturdays and Sundays. To keep momentum despite this irregular schedule, I broke the work into small, manageable tasks and tackled them one by one.

I began with the back-end and database design: sketching the ERD, normalizing tables into 3NF, defining Sequelize models, and writing migration scripts. Once the core API endpoints were in place (CRUD for products, categories, brands, users, carts, and orders, plus JWT-based authentication), I shifted to the front-end, building the admin interface page by page using Express, EJS, and Bootstrap. Each page “worked” end-to-

end on first pass, but I often looped back later to tweak responsive layouts, refine form validations, or integrate newly added API fields.

Reviewing my Git commit history served as a running log: I could see that the first weekend yielded a working database schema and initial CRUD services; by the second weekend, I had JWT middleware and user registration/login fully functional; by the third, the admin UI was up and I was polishing search and order-management views. Even when a feature seemed complete, I revisited it for small refinements. This iterative rhythm let me steadily advance the project until all requirements—Swagger docs, Jest/Supertest tests, README with .env example, and reflection report—were fulfilled.

2. Challenges faced during the development of this project

Time management. Finding uninterrupted coding blocks was my biggest hurdle. Evening slots were often just an hour or two, and I couldn't code every weeknight. The real push came on weekends: committing four to eight-hour sprints on Saturday and Sunday was essential. In hindsight, starting even earlier and sketching a minimal-viable-version in the first week would have reduced late-night marathons and last-minute tweaks.

Iterative front-end tweaks. While no single bug derailed me, the most frequent interruptions were small UI adjustments—tweaking a Bootstrap grid for the products table, repositioning error messages under form fields, or updating EJS templates after an API response change. Each pass of manual testing revealed tiny issues that required quick revisits, slowing the flow but improving polish.

Problem-solving resources. Whenever I hit a roadblock—like writing a raw SQL query for the /search endpoint or configuring role-based access in JWT middleware—I first consulted the course materials and official documentation. When I still felt stuck, I turned to ChatGPT with focused questions (for example, “How can I modularize my Express middleware for authentication and admin checks?”). Its suggestions were often spot-on, but I learned to verify each recommendation against official docs and my own logic, since AI can sometimes misinterpret context.

Planning vs. learning on the fly. I discovered that implementing a feature often led straight into the need to refactor once I learned a better approach—especially with Sequelize service patterns or EJS partials. This “learn-refactor” cycle meant revisiting code multiple times. While it lengthened the timeline, it also deepened my understanding and resulted in cleaner, more maintainable code.

Reusing past work. Looking ahead, for similar projects I plan to bootstrap the initial structure by reusing modules from previous assignments—database setup, authentication scaffolding, or test configurations. Even if requirements differ slightly, carrying over proven patterns can speed up the early phases dramatically.

By breaking the project into discrete chunks, leveraging AI judiciously, and embracing an iterative workflow—even when time was tight—I overcame every challenge without major delays and delivered a fully functioning back-end, API, admin front-end, test suite, and comprehensive documentation.