1. System architecture

   For start of project I choose monolithic architecture (Modular-monolothic) reas for that is can build fast that microservices and les complex. Also using modular monolithic it can be convert as microservices when it needed scale. For example customers growing or need multi region expansion.

$$\begin{tikzcd}\text{Client(Browser or Mobile)} \arrow[r, Rightarrow] & \text{HTTP Requests} \arrow[r, Rightarrow] & \text{Backend}\end{tikzcd}$$

   Then inside backend,
      Security Check (Authentication, Authorization),

   Controllers (thin controllers)

   Services (where business logic)

   Repository (Database access layer)

   Then after processing it return response to client.

2. Database Design

I choose no-sql db for this like MongoDB Atlas. Because using this I can reduce my manual work for scaling, bacukup and security when I need simple setup first. Also this has flexible schema that can use for when first phase of project. But we move to payment section it need well defined and solid variable because consider security.

$\boxed{\text{Users}}$ $\boxed{\text{Products}}$ $\boxed{\text{Orders}}$ $\boxed{\text{Audit logs}}$

$\boxed{\text{Customers}}$ $\boxed{\text{Cart}}$

Like these we can configure basic database collection first when it required field and optional fields,

User

- First_name
- Last_name
- Email
- Mobile
- isActive
- createdAt
- username

Products

- name
- category
- uom
- price
- quantity
- status
- last_stock_date

orders

- name
- customer – foreignkey
- date
- status

audit logs

- entity
- action
- prevValue
- newValue
- ip
- performdBy foriegnkey
- createdAt

Cart must be collection with temporary documents based userId.

3. I prefer separate backend end frontend because it easily scalable if we move microservis. And other optimization and code clarity also inceraesble.

4. Tech Stack
   Frontend – Next.js
   Backend – Nest.js
   DB – MongoDb
   Auth – Auth0(if need multifactor auth or SSO)

5. Cloud Infrastructure

I Choose AWS and EC2 for this and also S3 for images, if we need we can use cloudfront also performance.

6. Scalability

   Horizontal scaling with aws and mongodb support. And also need use load balancer.