

# Checkout Flow Interview Task (Frontend Intern - Next.js + TypeScript)

## 1. Objective

Implement a 3-step checkout flow from the provided Figma design:

- Account
- Shipping
- Payment

Figma reference: [Checkout-view \(Community\)](#)

Timebox: 6-8 hours (single sitting) or 1 day max.

## 2. What We Are Testing

- UI implementation quality (fidelity + responsive behavior)
- Ability to design clean forms and usable step-based flows
- API integration with a dummy backend built in Next.js
- Internationalization fundamentals (labels, currency formatting)
- Basic testing discipline

## 3. Required Implementation

### A. Checkout Screens (match Figma)

Implement 3 screens/steps:

1. Account
2. Shipping
3. Payment

Requirements:

- Progress/step indicator visible and accurate
- “Back” and “Next/Continue” controls
- Final action: “Complete order”
- Order summary card visible across steps (as shown in design)

- Mobile + desktop responsive behavior

## B. Form Fields + Validation

### Account step

- Email (required, valid email format)
- Password (required, min length 8)

### Shipping step

- First line of address (required)
- Street name (required)
- Postcode (required)
- Shipping method select (required)

### Payment step

- Name on card (required)
- Card number (required, numeric, basic length validation)
- Expiration month/year (required)
- CVC (required, 3-4 digits)

### Validation rules:

- Block step progression when invalid
- Show inline validation messages
- Preserve entered data when navigating between steps

## C. Dummy Backend with Next.js API Routes (mandatory)

Create route handlers under `src/app/api/checkout/*` (or `app/api/checkout/*`):

- GET `/api/checkout/summary`
- POST `/api/checkout/account`
- POST `/api/checkout/shipping`
- POST `/api/checkout/payment`
- POST `/api/checkout/complete`

### Requirements:

- Return realistic mock payloads
- Simulate network latency (400–800ms)
- Include both success and error responses
- Frontend must consume these endpoints (no hardcoded final results in UI)

#### D. API Integration

- Use the API routes for submit/fetch actions
- Handle loading states (buttons, form actions)
- Handle errors with visible messages
- Prevent duplicate submits while request is in progress

#### E. Internationalization (required)

Support at least two languages/locales:

- English + one additional language of your choice

Must localize:

- Checkout labels/buttons/messages (simple dictionary approach is fine)
- Currency in order summary using locale-aware formatting

#### F. Package Manager

- Use npm for dependency management and scripts.

### 4. Deliverables

- Working implementation of all 3 checkout steps
- Dummy API route handlers wired to UI
- Responsive layout matching the reference design
- Loading/error/validation states

- [README\\_ASSIGNMENT.md](#) including:

## 5. Hard Constraints (Fail if violated)

- Must implement all 3 Figma checkout steps (Account, Shipping, Payment)
- Must use Next.js API route handlers as dummy backend
- Must integrate frontend with those handlers
- Must support two languages/locales (English + one additional language)
- Must include validation + loading + error states
- Must use pnpm

## 6. Scoring Rubric (100)

- Design fidelity + responsiveness: 35
- API route design + frontend integration: 25
- Form flow + validation quality: 20
- i18n correctness: 10
- Code quality, documentation: 10