# Take-Home Challenge — Frontend Development Roadmap

This document outlines a step-by-step roadmap for developing a reusable, dataset-agnostic Filter Builder UI library. The library allows users to construct arbitrary nested conditions (and/or groups) and serialize them into a JSON structure in the required format. It also covers schema-driven configuration, validation rules, API integration, UX, and testing.

## Phase 1: Foundations

* - Setup project with Vite + React + TypeScript.
* - Create monorepo or single repo with /packages/filter-builder (library) and /examples/demo-app (usage).
* - Install testing tools (Vitest/Jest).
* - Define domain types: FieldDef, Condition, GroupNode, OperatorKey, etc.

## Phase 2: Core Logic

* - Implement operator registry with validation rules (arity: none, one, two, array).
* - Add validation engine for rules (between, in, is\_null, etc.).
* - Create serializer functions: fromWire (load {and|or} JSON) and toRequiredWire (export JSON).

## Phase 3: UI Components

* - Build FilterBuilder React component (recursive UI).
* - Group component: AND/OR toggle, add/remove groups.
* - Condition component: field select, operator select, type-aware value input.
* - Validation feedback UI.

## Phase 4: API Integration

* - Transport layer: buildGetRequest, buildPostRequest.
* - Configurable props: endpoint, GET/POST mode.
* - Emit events via onChange with JSON and query string.

## Phase 5: UX & Accessibility

* - Keyboard navigation and ARIA labels for accessibility.
* - Responsive layout using CSS grid/flex.
* - Import/Export JSON (textarea, copy button) in example app.

## Phase 6: Testing

* - Unit tests for serialization/deserialization, validation, GET/POST.
* - Integration tests for add/remove/edit conditions and groups.
* - Verify onChange emits correct JSON.

## Phase 7: Deliverables

* - Library source code.
* - Example app with at least 2 datasets (Users, Products).
* - README with installation, usage, configuration, and architecture notes.

## Estimated Timeline

• Experienced React/TypeScript developer: 1–1.5 weeks full-time.  
• Developer learning React/TS/validation patterns: 2–3 weeks.