Application Failure Analysis — Sample

Candidate: Alexander Cruz
Role Target: Application Analyst

---

1. Problem

Configurator API POST /api/configurator/price endpoint returned 400 Bad Request with error: "The JSON value could not be converted to Configurator.Core.Enums.ProductType". Failure occurred during JSON deserialization in request handling pipeline. Observable symptom: HTTP 400 response with JSON deserialization exception when enum values in request payload used lowercase or mixed-case formatting (e.g., "fancoil" instead of "FanCoil").

2. Root Cause

System.Text.Json default enum converter performed case-sensitive string-to-enum conversion. Request payloads with enum values in non-PascalCase format (e.g., "fancoil", "FANCOIL", "FanCoil") failed deserialization despite representing valid enum members. Missing case-insensitive enum conversion configuration in Program.cs JSON serializer options.

3. Resolution

Implemented CaseInsensitiveEnumConverter<T> class using Enum.TryParse with ignoreCase: true parameter. Registered converter factory in Program.cs via JsonSerializerOptions.Converters. Verified fix by sending test requests with lowercase enum values; API now accepts "fancoil", "FANCOIL", and "FanCoil" formats and returns 200 OK with correct pricing calculations.

4. Evidence

Screenshot 1: API Error Response

HTTP 400 response body showing: {"error":"The JSON value 'fancoil' could not be converted to ProductType"}

Proves: Enum deserialization failure with case-sensitive validation

Screenshot 2: Fix Implementation

CaseInsensitiveEnumConverter.cs lines 28-31: Enum.TryParse<T>(value, ignoreCase: true, out var result)

Proves: Case-insensitive enum parsing logic implemented

Screenshot 3: Successful Request After Fix

HTTP 200 response with payload: {"productType":"fancoil",...} → {"configurationId":"CFG-...","unitPrice":...}

Proves: Lowercase enum values now accepted and processed correctly