# Exception Handling & Centralized Logging Documentation

## 1. Overview

This document outlines the implementation of a robust centralized logging and exception handling mechanism for a Spring Boot Java application using SLF4J with Logback. It includes a detailed guide on log format customization, trace ID correlation, error response generation, and structured error logging.

## 2. Components Overview

Key components involved in the implementation:

* • SLF4J (Logging facade)
* • Logback (Logging backend)
* • GlobalExceptionHandler (Centralized exception management)
* • TraceIdFilter (Trace ID generator and MDC setup)
* • ErrorHandlingUtil (Error mapping utilities)

## 3. SLF4J and Logback Configuration

SLF4J (Simple Logging Facade for Java) provides a generic interface for logging. Logback is the chosen implementation backend. Logback is configured programmatically to include enhanced formatting and traceability.

The encoder pattern used:

%cyan([%d{yyyy-MM-dd HH:mm:ss.SSS}]) %highlight(%-5level) [traceId=%X{traceId}] %blue([projectName]) %cyan(%file -> %class.%M) :%line %yellow(errorCode=%X{errorCode}) %msg%n

## 4. Log Color Mapping

|  |  |
| --- | --- |
| Log Level | Color |
| INFO | 🟢 Green |
| ERROR | 🔴 Red |
| WARN | 🟠 Yellow |
| DEBUG | 🔵 Blue |
| TRACE | ⚪ Gray |

## 5. TraceIdFilter

A servlet filter (`TraceIdFilter`) is used to generate and bind a `traceId` to the request thread using MDC (Mapped Diagnostic Context). This ID is injected into logs to track request flows across the application lifecycle.

## 6. GlobalExceptionHandler

This class handles all unhandled exceptions across the application and returns structured JSON error responses. It maps various exceptions like `HttpMediaTypeNotSupportedException`, `MethodArgumentNotValidException`, `ApiCustomException`, etc., into consistent error structures with helpful metadata.

## 7. Error Response Structure

Sample Error Response JSON:

{  
 "refId": "1a2b3c4d-1234-5678-9012-abcdef123456",  
 "code": "ERR\_INVALID\_REQUEST",  
 "message": "Request validation failed.",  
 "severity": "ERROR",  
 "status": 400,  
 "errorDetails": [  
 {  
 "field": "email",  
 "message": "Email format is invalid."  
 }  
 ]  
}

## 8. Error Code Mapping

All error codes are mapped to user-friendly messages via `ErrorHandlingUtil`. These are loaded from a centralized properties file or config. Fallback to a generic message occurs if no mapping is found.

## 9. Benefits of this Setup

* • Centralized error visibility and debugging
* • Consistent log format across environments
* • Seamless integration of `traceId` in logs for distributed tracing
* • Enhanced developer experience during debugging and testing