High-Level Design Document

# Online Judge Platform

Prem Arora

Updated: June 23, 2025

# 1. Introduction

This document outlines the high-level design (HLD) for an Online Judge (OJ) platform built using the MERN stack with Docker-based backend evaluation. The platform allows users to solve programming problems and get real-time feedback through automated code evaluation, similar to platforms like HackerRank and LeetCode.

# 2. Key Features

* • User Authentication via JWT and bcrypt
* • Role-based access (Admin/User)
* • Problem creation, editing, and categorization by Admins
* • Integrated online code editor with language selection
* • Docker-powered code execution and automated verdicts
* • Real-time submission feedback and execution stats
* • Leaderboard showcasing top performers
* • Difficulty-wise categorization (Easy, Medium, Hard)
* • Dark/Light Mode Toggle
* • Detailed User Dashboard with Charts

# 3. Database Design (MongoDB)

## Users Collection

{  
 \_id: ObjectId,  
 username: String,  
 email: String,  
 passwordHash: String,  
 role: String, // "admin" or "user"  
 createdAt: Date,  
 lastLogin: Date  
}

## Problems Collection

{  
 \_id: ObjectId,  
 title: String,  
 code: String, // Unique code  
 description: String,  
 difficulty: "Easy" | "Medium" | "Hard",  
 timeLimit: Number, // in seconds, moved from TestCases  
 createdAt: Date  
}

## Solutions Collection

{  
 \_id: ObjectId,  
 userId: ObjectId,  
 problemId: ObjectId,  
 code: String,  
 language: String,  
 verdict: String,  
 submittedAt: Date  
}

## TestCases Collection

{  
 \_id: ObjectId,  
 problemId: ObjectId,  
 input: String,  
 expectedOutput: String,  
 isHidden: Boolean  
}

# 4. User Interface Design

* • Home Page - Shows problems with filters by difficulty
* • Problem Page - Displays description, editor, language dropdown, and submission button
* • Verdict Page - Shows output and past submission history
* • Leaderboard - Lists top scorers by time frame
* • Dashboard - (Optional) Displays user stats in chart form using Chart.js or Recharts

# 5. Backend API Endpoints

* • GET /api/problems - Get all problems
* • GET /api/problems/:id - Get problem by ID
* • POST /api/submit - Submit code for evaluation
* • POST /api/signup - Register new user (default role: user)
* • POST /api/login - Authenticate user
* • GET /api/leaderboard - Fetch top submissions

# 6. Evaluation Engine

Code is executed in Docker containers for security and isolation. Each submission is tested against public and hidden test cases. Verdicts are generated based on output comparison.

# 7. Planned Enhancements

* • Contest mode with countdown timer
* • Gamified achievements and badges
* • Detailed charts for submission trends
* • Editor themes (Dark/Light Toggle)
* • Admin panel with CRUD for problems and users