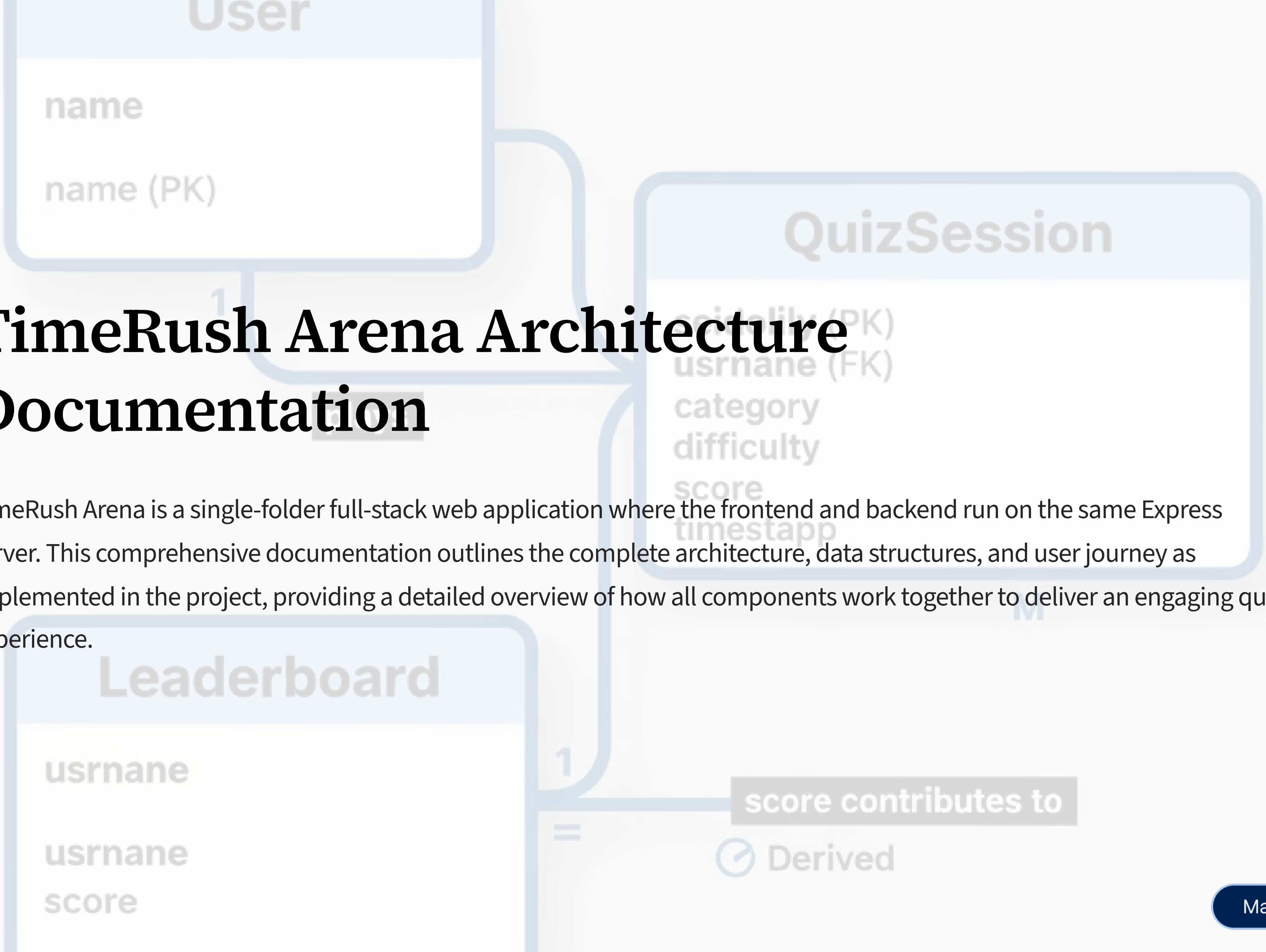
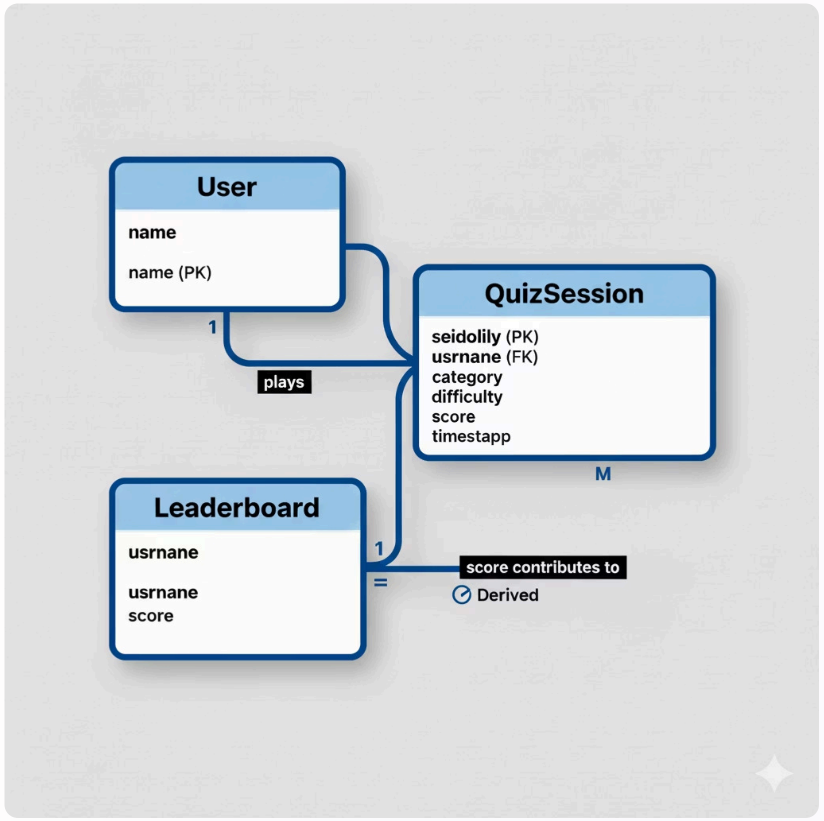


# TimeRush Arena Architecture Documentation

TimeRush Arena is a single-folder full-stack web application where the frontend and backend run on the same Express server. This comprehensive documentation outlines the complete architecture, data structures, and user journey as implemented in the project, providing a detailed overview of how all components work together to deliver an engaging quiz experience.



# Entity Relationship Diagram

The ERD illustrates the actual data entities and their relationships within the TimeRush Arena project. The system comprises three core entities that work together to deliver the quiz experience.

## User

Users are identified solely by their name with no authentication required in the MVP version.

- name (identifier)
- No password needed
- Simple entry system

## QuizSession

Each gameplay creates a unique quiz session with comprehensive tracking.

- sessionId
- userName
- category & difficulty
- score & timestamp

## Leaderboard

The leaderboard stores rankings in backend memory for quick access.

- userName
- score
- In-memory storage

# Entity Relationships

The relationships between entities define how data flows and connects throughout the application. Understanding these connections is crucial for maintaining data integrity and ensuring seamless gameplay experiences.

## User to QuizSession

One user can play multiple quiz sessions, allowing for repeated gameplay and score improvement.

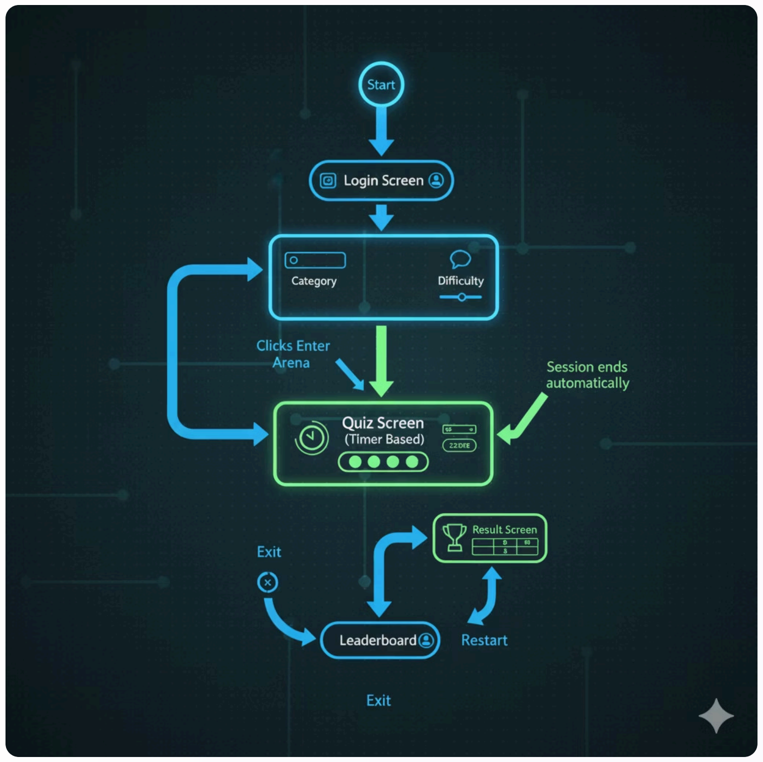
## QuizSession to Score

Each quiz session generates exactly one score that represents the player's performance.

## Score to Leaderboard

Quiz session scores contribute to the leaderboard, creating competitive rankings.

## Data Flow Architecture



The Data Flow Diagram illustrates the actual movement of data between the frontend, backend, and external APIs. This architecture ensures seamless communication across all system components.

01

### User Opens Application

Browser loads the application from the Express server.

02

### Frontend Loads

Static assets and interface elements are served to the client.

03

### Category Selection

User chooses quiz category and difficulty level.

04

### API Request

Frontend calls backend API endpoint /api/questions.

05

### External Data Fetch

Backend retrieves questions from Open Trivia API.

06

### Questions Delivered

Backend sends formatted questions to frontend.

## Score Processing Flow

After users complete the quiz, their performance data follows a specific path through the system to update rankings and maintain the leaderboard.

### User Interaction

Users answer quiz questions within the timed session. The frontend tracks responses and calculates the final score based on correct answers.

- Questions answered
- Time tracking
- Score calculation

### Backend Processing

The score is transmitted to the backend via /api/score endpoint. The system updates the in-memory leaderboard and makes rankings available through /api/leaderboard.

- Score submission
- Leaderboard update
- Rankings retrieval

## User Journey Overview

The user journey maps the complete experience from application launch to game completion. Each screen serves a specific purpose in guiding users through the quiz experience.

1. **Application Start**

   User launches TimeRush Arena in their browser.

2. **Login Screen**

   User enters their name for identification.

3. **Home Screen**

   Category and difficulty selection interface.

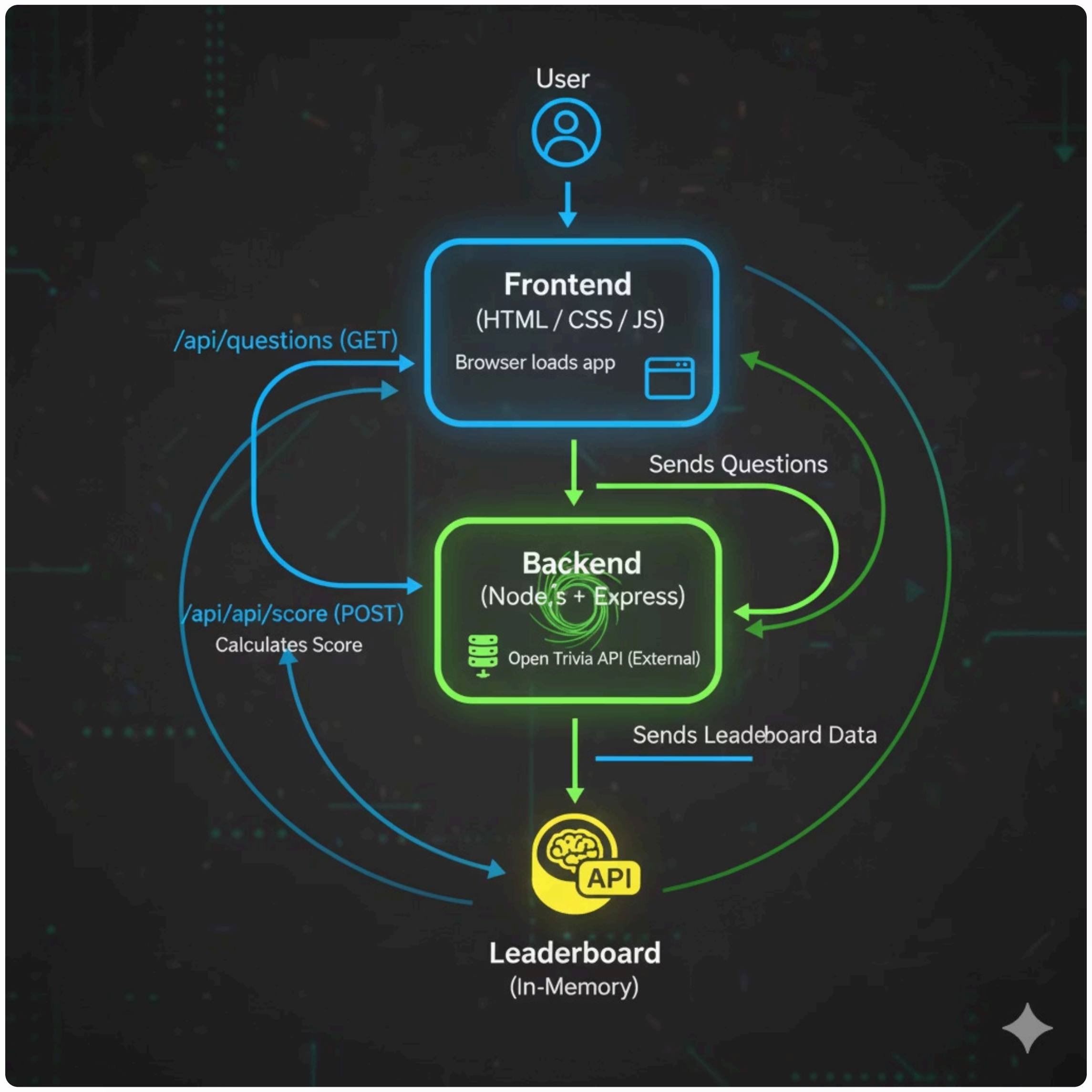
4. **Enter Arena**

   User clicks button to begin the quiz session.

5. **Quiz Screen**

   Timed questions appear with countdown timer.

# Quiz Experience Flow



The core quiz experience is designed for engagement and challenge. Users navigate through timed questions with immediate feedback and comprehensive results.

### Timed Questions

Users answer questions within the allocated time limit. The timer creates urgency and excitement throughout the session.

### Answer Submission

Each answer is recorded and validated. The system tracks correct responses for final score calculation.

### Session Completion

The quiz ends automatically when time expires. Users are immediately shown their performance results.

## Results and Leaderboard

After completing the quiz, users receive comprehensive feedback on their performance and can compare their scores with other players on the leaderboard.

### Result Screen

Displays the user's final score with detailed breakdown of correct and incorrect answers.

### Leaderboard Display

Shows rankings of all players, allowing users to see how they compare with others.

### Exit or Restart

Users can choose to exit the application or start a new quiz session to improve their score.

## Technology Stack

TimeRush Arena utilises a streamlined technology stack that enables rapid development and deployment. The single-folder architecture simplifies maintenance and hosting.

### Frontend Technologies

Built with HTML, CSS, and JavaScript for a responsive and interactive user interface. The frontend handles all user interactions and quiz logic.

### Backend Framework

Node.js with Express powers the backend, serving both static files and API endpoints from a single server instance.

### Data Management

Open Trivia Database provides quiz questions whilst the leaderboard uses in-memory storage for fast access and simple deployment.

## Architecture Summary

The complete architecture demonstrates a well-integrated system where all components work together seamlessly. This single-folder, single-server approach provides simplicity without sacrificing functionality.

| Layer | Implementation |
|---|---|
| Frontend | HTML, CSS, JavaScript |
| Backend | Node.js + Express |
| API | Open Trivia Database |
| Storage | In-memory leaderboard |
| Deployment | Single-folder, single-server |

This architecture enables TimeRush Arena to deliver a fast, engaging quiz experience whilst maintaining simplicity in development and deployment.