**End-User Documentation Handout**

**Introduction**

Welcome to the Quiz Platform! This platform is designed to enhance the learning experience for students, teachers, and administrators. It provides a seamless environment for creating, managing, and taking quizzes, tracking progress, and much more.

**Website Features**

**1. User Roles**

* **Admin**: Manage users, quizzes, view dashboard, and stats.
* **Teacher**: Add questions, view student answers, and manage quizzes.
* **Student**: Take lessons, attempt quizzes, and view progress.

**2. User Accounts**

* **Student Accounts**: Created through self-registration.
* **Teacher Accounts**: Created by the admin.

**3. Quizzes & Lessons**

* **Creating Quizzes**: Admin/Teachers can create quizzes with image-based MCQs.
* **Taking Quizzes**: Students can take quizzes related to lessons.
* **Lock Mechanism**: Quizzes unlock based on previous test results (> 80% score).
* **Multiple Tests**: Multiple tests may correspond to a single lesson.

**4. Progress Tracking**

* **Student View**: See lessons, quizzes, and overall progress.
* **Teacher/Admin View**: Monitor individual and overall student progress.

**5. Email Notifications**

* **Students**: Receive test results via email.
* **Admin**: Optionally receive detailed test results.

**6. Website Stats**

* **Dashboard**: Display the number of students, quizzes, average scores, etc.

**7. Hosting & Deployment**

* **Platform**: Hosted on a public hosting platform like Digital Ocean.

**User Guides**

**For Students**

Registration & Login

* **Register**: Click on "Register" and fill in your details.
* **Login**: Use your credentials to log in and start your lessons.

Taking Quizzes

* **Access Quizzes**: Access available quizzes from the dashboard.
* **Answering Questions**: Select the correct answer from the multiple-choice options.
* **Test Results**: View your test results immediately or receive them via email.

**For Teachers**

Managing Quizzes

* **Create Quizzes**: Use the "Create Quiz" option to set up new quizzes.
* **View Student Answers**: Access detailed student answers from your dashboard.

**For Admins**

User Management

* **Create Teacher Accounts**: Use the admin panel to create teacher accounts.
* **Manage Users**: View and manage all user accounts.

Quiz & Stats Management

* **Manage Quizzes**: Create, edit, and delete quizzes.
* **View Stats**: Access the dashboard for an overview of platform statistics.

**Conclusion**

This platform aims to provide a robust, engaging, and effective learning experience. From self-paced lessons to challenging quizzes, it offers a comprehensive educational environment. For further assistance or queries, please contact the support team.

T**able of Contents**

1. Introduction
2. System Architecture
3. Database Schema
4. API Endpoints
5. Authentication & Authorization
6. Frontend Implementation
7. Error Handling & Logging
8. Configuration & Environment Management
9. Deployment & Hosting
10. Conclusion

**1. Introduction**

**The Quiz App is an educational platform that facilitates quiz creation, management, participation, and tracking. This document provides comprehensive technical details on the system's architecture, database design, API endpoints, authentication, and more.**

**2. System Architecture**

**The system architecture is divided into frontend and backend components.**

**Backend:**

* **Framework: Flask**
* **Database: SQLite**
* **ORM: SQLAlchemy**

**Frontend:**

* **Framework: React**

**3. Local Environment Setup**

**Setting up the application locally involves configuring both the backend and frontend.**

**Backend Setup:**

1. **Clone the Repository: Download the code from the repository.**
2. **Navigate to Server Directory: Open a terminal and navigate to the server directory.**
3. **Install Dependencies: Run the command pip install -r requirements.txt to install the required packages.**
4. **Start the Server: Execute python app.py to start the backend server.**

**Frontend Setup:**

1. **Navigate to Frontend Directory: Open a terminal and navigate to the frontend directory.**
2. **Install Dependencies: Run the command npm install to install required packages.**
3. **Start the Server: Execute npm start to start the frontend server.**

**4. Database Schema**

**The database schema consists of multiple tables. Here's an example of the Question model:**

**pythonCopy code**

**class Question(Base): \_\_tablename\_\_ = "questions" id = Column(Integer, primary\_key=True) content = Column(JSON) quiz\_id = Column(Integer, ForeignKey("quizzes.id")) created\_date = Column(DateTime, default=func.now(), nullable=False) updated\_date = Column(DateTime, default=func.now(), onupdate=func.now(), nullable=False)**

**Save to grepper**

**5. API Endpoints**

**The application exposes several API endpoints. Here's the implementation for uploading a quiz:**

**pythonCopy code**

**@quiz.route("/upsert\_quiz", methods=["POST"]) def upload\_quiz(): try: # logic for uploading a quiz except Exception as e: traceback.print\_exc() return jsonify({"error": str(e)}), 500**

**Save to grepper**

**6. Authentication & Authorization**

**Authentication is implemented using JWT (details and code snippets needed for this section).**

**7. Frontend Implementation**

**The frontend is structured with various components. For example, the Quiz component might look like:**

**javascriptCopy code**

**function Quiz() { // logic to render a quiz }**

**Save to grepper**

**8. Error Handling & Logging**

**Error handling and logging are crucial for diagnosing issues. The application uses Python's built-in logging and exception handling:**

**pythonCopy code**

**import logging logger = logging.getLogger("auth") try: # logic except Exception as e: traceback.print\_exc() logger.error(str(e))**

**Save to grepper**

**9. Deployment & Hosting**

**The application can be hosted on platforms like Digital Ocean. Details of the deployment process and server configuration will be specific to the chosen hosting platform.**

**10. Conclusion**

**This document provides a detailed technical overview of the Quiz App, covering aspects from system architecture to code-level implementation. It serves as a comprehensive guide for developers, system administrators, and other technical stakeholders involved in the application.**