Java code for console:

import java.util.\*;

public class QuizPlatform {

    static List<User> users = new ArrayList<>();

    static List<Category> categories = new ArrayList<>();

    static List<Quiz> quizzes = new ArrayList<>();

    static List<Question> questions = new ArrayList<>();

    public static void main(String[] args) {

        initializeMockData();

        Scanner scanner = new Scanner(System.in);

        System.out.println("--- Add New Quiz ---");

        System.out.print("Enter quiz title: ");

        String title = scanner.nextLine();

        System.out.print("Enter category (1. General Knowledge, 2. Science, 3. Mathematics): ");

        int categoryId = scanner.nextInt();

        Category selectedCategory = categories.get(categoryId - 1);

        System.out.print("Enter difficulty level (1-5): ");

        int difficultyLevel = scanner.nextInt();

        Quiz newQuiz = new Quiz(quizzes.size() + 1, title, selectedCategory, difficultyLevel, new Date());

        quizzes.add(newQuiz);

        System.out.println("Quiz added: " + newQuiz);

        scanner.nextLine();

        while (true) {

            System.out.println("\n--- Add New Question ---");

            System.out.print("Enter question text: ");

            String questionText = scanner.nextLine();

            System.out.print("Enter correct answer: ");

            String correctAnswer = scanner.nextLine();

            System.out.print("Enter wrong answers (comma separated): ");

            String wrongAnswers = scanner.nextLine();

            Question newQuestion = new Question(questions.size() + 1, questionText, newQuiz, selectedCategory, correctAnswer, wrongAnswers);

            questions.add(newQuestion);

            System.out.println("Question added: " + newQuestion);

            System.out.print("\nDo you want to add another question? (y/n): ");

            String continueInput = scanner.nextLine();

            if (!continueInput.equalsIgnoreCase("y")) {

                break;

            }

        }

        addIndiaQuizQuestions(newQuiz);

        checkAnswer(scanner);

        displayData();

    }

    private static void initializeMockData() {

        categories.add(new Category(1, "General Knowledge"));

        categories.add(new Category(2, "Science"));

        categories.add(new Category(3, "Mathematics"));

        Quiz scienceQuiz = new Quiz(1, "Science Quiz", categories.get(1), 2, new Date());

        quizzes.add(scienceQuiz);

    }

    private static void addIndiaQuizQuestions(Quiz quiz) {

        questions.add(new Question(1, "What is the capital of India?", quiz, categories.get(0), "Delhi", "Mumbai, Bangalore, Kolkata"));

        questions.add(new Question(2, "Who is the first Prime Minister of India?", quiz, categories.get(0), "Jawaharlal Nehru", "Mahatma Gandhi, Sardar Patel, Indira Gandhi"));

        questions.add(new Question(3, "Which is the largest state in India by area?", quiz, categories.get(0), "Rajasthan", "Uttar Pradesh, Madhya Pradesh, Maharashtra"));

        questions.add(new Question(4, "Which Indian city is known as the 'Silicon Valley of India'?", quiz, categories.get(0), "Bangalore", "Chennai, Hyderabad, Pune"));

        questions.add(new Question(5, "What is the national animal of India?", quiz, categories.get(0), "Tiger", "Elephant, Lion, Peacock"));

    }

    private static void checkAnswer(Scanner scanner) {

        System.out.println("\n--- Answer Checking ---");

        System.out.print("Enter the quiz ID to answer: ");

        int quizId = scanner.nextInt();

        Quiz selectedQuiz = null;

        for (Quiz quiz : quizzes) {

            if (quiz.getQuizId() == quizId) {

                selectedQuiz = quiz;

                break;

            }

        }

        if (selectedQuiz == null) {

            System.out.println("Quiz not found!");

            return;

        }

        System.out.println("Now answering questions for quiz: " + selectedQuiz.getTitle());

        scanner.nextLine();

        List<Question> selectedQuestions = new ArrayList<>();

        for (Question question : questions) {

            if (question.getQuiz().equals(selectedQuiz)) {

                selectedQuestions.add(question);

            }

        }

        if (selectedQuestions.isEmpty()) {

            System.out.println("No questions found for this quiz.");

            return;

        }

        for (Question question : selectedQuestions) {

            System.out.println("Question: " + question.getQuestionText());

            List<String> options = question.getOptions();

            Collections.shuffle(options);

            for (int i = 0; i < options.size(); i++) {

                System.out.println((i + 1) + ". " + options.get(i));

            }

            System.out.print("Your answer (1-" + options.size() + "): ");

            int userAnswer = scanner.nextInt();

            if (options.get(userAnswer - 1).equalsIgnoreCase(question.getCorrectAnswer())) {

                System.out.println("Correct!");

            } else {

                System.out.println("Incorrect! The correct answer is: " + question.getCorrectAnswer());

            }

        }

    }

    private static void displayData() {

        System.out.println("\n--- All Quizzes and Questions ---");

        for (Quiz quiz : quizzes) {

            System.out.println(quiz);

            for (Question question : questions) {

                if (question.getQuiz().equals(quiz)) {

                    System.out.println("\t" + question);

                }

            }

        }

    }

}

class User {

    private int userId;

    private String name;

    private String email;

    private String password;

    private String profilePicture;

    private Date accountCreated;

    public User(int userId, String name, String email, String password, String profilePicture, Date accountCreated) {

        this.userId = userId;

        this.name = name;

        this.email = email;

        this.password = password;

        this.profilePicture = profilePicture;

        this.accountCreated = accountCreated;

    }

    @Override

    public String toString() {

        return "User{id=" + userId + ", name='" + name + "', email='" + email + "'}";

    }

}

class Category {

    private int categoryId;

    private String categoryName;

    public Category(int categoryId, String categoryName) {

        this.categoryId = categoryId;

        this.categoryName = categoryName;

    }

    @Override

    public String toString() {

        return "Category{id=" + categoryId + ", name='" + categoryName + "'}";

    }

}

class Quiz {

    private int quizId;

    private String title;

    private Category category;

    private int difficultyLevel;

    private Date createdAt;

    public Quiz(int quizId, String title, Category category, int difficultyLevel, Date createdAt) {

        this.quizId = quizId;

        this.title = title;

        this.category = category;

        this.difficultyLevel = difficultyLevel;

        this.createdAt = createdAt;

    }

    public int getQuizId() {

        return quizId;

    }

    public String getTitle() {

        return title;

    }

    @Override

    public String toString() {

        return "Quiz{id=" + quizId + ", title='" + title + "', category=" + category + ", difficultyLevel=" + difficultyLevel + "}";

    }

}

class Question {

    private int questionId;

    private String questionText;

    private Quiz quiz;

    private Category category;

    private String correctAnswer;

    private String wrongAnswers;

    public Question(int questionId, String questionText, Quiz quiz, Category category, String correctAnswer, String wrongAnswers) {

        this.questionId = questionId;

        this.questionText = questionText;

        this.quiz = quiz;

        this.category = category;

        this.correctAnswer = correctAnswer;

        this.wrongAnswers = wrongAnswers;

    }

    public Quiz getQuiz() {

        return quiz;

    }

    public String getQuestionText() {

        return questionText;

    }

    public String getCorrectAnswer() {

        return correctAnswer;

    }

    public List<String> getOptions() {

        List<String> options = new ArrayList<>();

        options.add(correctAnswer);

        String[] wrongAnswersArray = wrongAnswers.split(",");

        for (String answer : wrongAnswersArray) {

            options.add(answer.trim());

        }

        return options;

    }

    @Override

    public String toString() {

        return "Question{id=" + questionId + ", text='" + questionText + "', correctAnswer='" + correctAnswer + "', wrongAnswers='" + wrongAnswers + "'}";

    }

}

