Author: Negoescu-Cîrstea Ștefan-Daniel

Online platform for knowledge testing

Specification of basic and additional functionalities

A. Basic Functionalities

1. **Adding Questions**:
    - It allows the user to add a new question to the quiz.
    - Support for multiple-choice questions:
        - A correct answer.
        - More wrong answers.
2. **Deleting and editing questions**:
    - The ability to delete/modify a question based on the numerical index.
    - Validation for user input (e.g. whether it is a valid number and matches an existing question).
3. **Viewing the quiz**:
    - Listing all the added questions.
    - View the right and wrong answers associated with each question.

B. Additional functionalities

1. **Error and success messages**:
    - Displaying descriptive messages for the user (e.g. "The question was successfully added/deleted", "The number entered is invalid").
2. **Making the quiz with questions and answers in random order**
    - The user can solve a quiz where:
        - The questions are displayed in a random order.
        - Each question provides several possible answers (including the correct one), and they are displayed in a random order.
        - The user selects the answers that they think are correct using checkboxes.

- After completing the quiz, the score (the number of questions answered correctly out of the total) is displayed.

3. User-friendly graphical interface:

- Intuitive forms for adding and deleting questions.
- Differentiated visual messages for success (green) and error (red).

4. Easy Navigation Between Pages:

- Clear links to switch between features (e.g. "View quiz").

Description of project classes/methods/attributes

Class

1. NegoescuDanielApplication
   - **Description:** The *NegoescuDanielApplication class* is the main class of the Spring Boot application. It initializes and launches the application using the Spring Boot framework.
   - **Methods**
     - **main(String[] args)** Starts the application by calling SpringApplication.run. Create and configure the Spring Boot context.
2. QuizController
   - **Description:** The *QuizController* class is responsible for managing the application logic for quiz-related operations, including adding, modifying, deleting questions, as well as the functionality of quiz completion by users.
   - **Attributes**
     - **Question List<Question> Questions** store questions to display, modify, or delete.
   - **Methods**
     - The **QuizController()** constructor initializes the list of questions with predefined examples, including single-answer questions and multiple-choice questions.
     - **Methods for Page Management**
       - **@GetMapping("/view_quiz") viewQuiz(Model model)** displays all the questions in the quiz.
       - **@GetMapping("/") index()** redirects to the main page.
       - **@GetMapping("/add_question") addQuestion()** redirects the user to the form for adding a question.
       - **@GetMapping("/modify_question") modifyQuestion()** displays the form for modifying questions.

        - **@GetMapping("/delete_question") deleteQuestion()** displays the form for deleting questions.
    - **Add-on features**
        - **@PostMapping("/add_question_sa") addQuestionSA(...)** Allows you to add a single-answer question.
        - **@PostMapping("/add_question_ma") addQuestionMA(...)** Allows you to add a multiple-choice question.
    - **Functionality for Modification**
        - **@PostMapping("/modify_question") modifyQuestion(...)** Changes the text of an existing question based on the index.
    - **Delete features**
        - **@PostMapping("/delete_question") deleteQuestion(...)** Changes the text of an existing question based on the index.
    - **Quiz functionality**
        - **@GetMapping("/start_quiz") startQuiz(Model model, @ModelAttribute("quizSession") QuizSession quizSession)** shuffles the questions and initializes the quiz session and sets the current index and user score to zero.
        - **@GetMapping("/question") showQuestion(...)** Displays the current question, along with the available (mixed) options.
        - **@PostMapping("/submit_answer") submitAnswer(...)** Receives the user's answers and validates their correctness; updates the score if the answers are correct; moves on to the next question or redirects to the final result.
        - **@GetMapping("/quiz_result") showQuizResult(...)** Displays the user's final score and total number of questions.
    - **Redirection Functionalities**
        - **@PostMapping("/choose-option") chooseOption(...)** redirects the user to different functionalities based on the selected option.
        - **@PostMapping("/choose-option1") chooseOption1(...)** Redirects to pages for viewing, adding, modifying, or deleting questions.
        - **@PostMapping("/choose-option2") chooseOption2(...)** Redirect to the forms for adding single or multiple choice questions.

3. QuizSession

- **Description:** The class is designed to preserve the state of the session for the duration of a quiz.

- **Attributes**
  - **Questions** is a list of *Question objects*, which represent the questions in the quiz.
  - **currentindex** is a whole that keeps track of the index of the current question in the list.
  - **Score** is an integer that represents the user's score currently.
- **Methods**
  - **Getters and Setters** provide access to private fields and allow them to be updated.
  - **incrementIndex()** increments the currentindex field to move on to the next question in the quiz.
  - **incrementScor()**: Increments the score field, usually when the user answers a question correctly.

4. Question

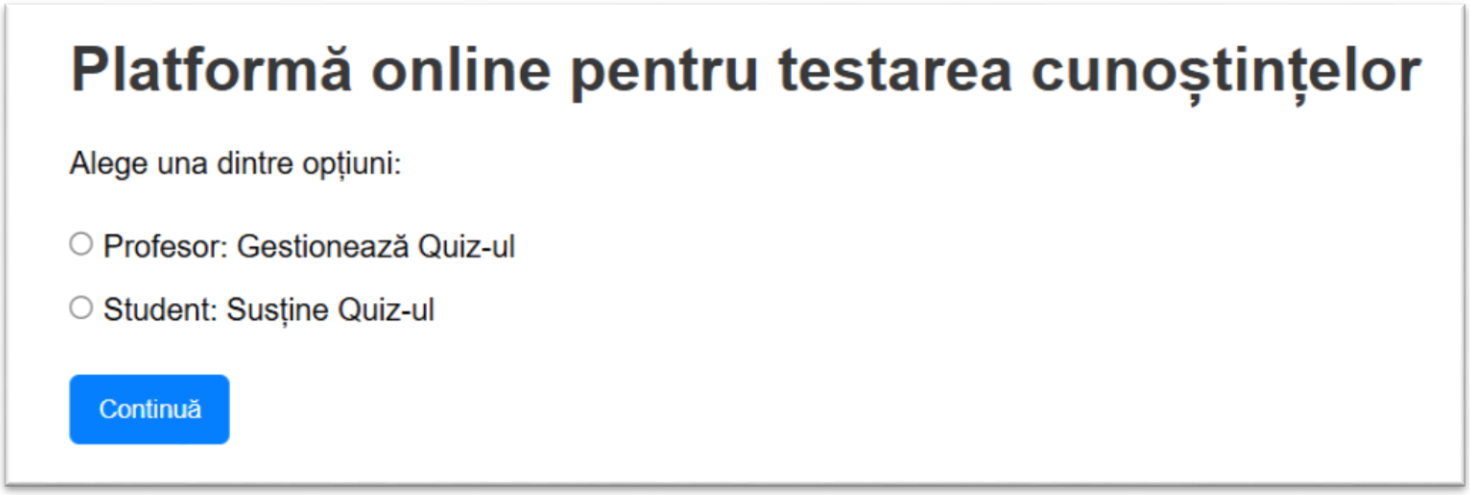
- **Description:** The *Question Class* is a generic question in a quiz. It is used as the basis for other, more specialized types of questions, which may have a single answer or multiple answers.
- **Attributes**
  - **question:** A variable of type String, which stores the text of the question.
- **Methods**
  - **Question(Question String)** The class constructor receives a text for the question and sets it as the value for the question attribute.
  - **getQuestion()** Returns the text of the question.
  - **setQuestion(Question String)** Allows you to modify the question.
  - **toString()** Overriding the toString() method allows you to obtain a textual representation of the object in the form of Question: [question text].

5. QuestionWithAnswerOnly / QuestionWithAnswerMultiple

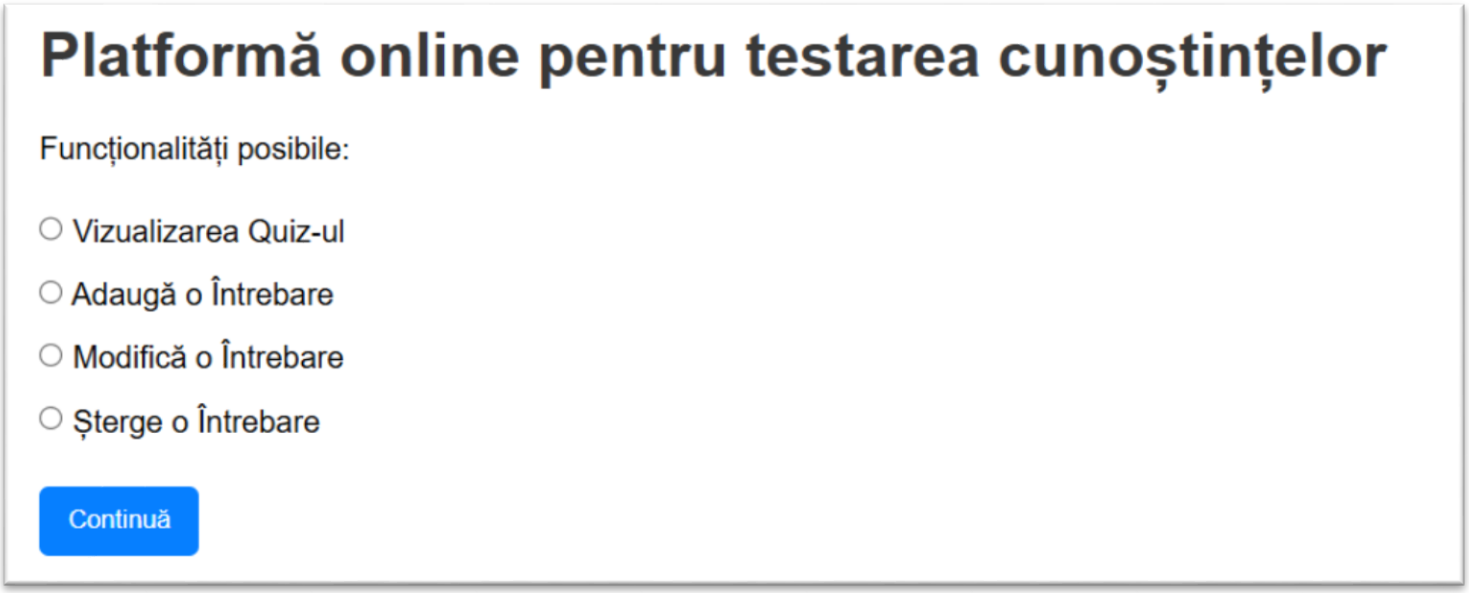
- **Attributes**
  - **answerCorrect:** A String that stores the correct answer to the question.
- **Methods**
  - **QuestionWithUniqueAnswer(String Question, String AnswerCorrect, List<String> WrongAnswers):** The constructor**.**
  - **get getAnswerCorrect()**: Returns the correct answer.
  - **setAnswerCorrect(String answerCorrect)**: Allows you to change the correct answer.
  - **setIncorrectAnswers(List<String> Incorrect Answers):** Allows you to modify the list of incorrect answers.
  - **getWrongAnswers()**: Returns the list of wrong answers.
  - **toString():** Overrides the toString() method in the parent Question class and adds details about right and wrong answers.

- **getPossibleAnswers()**: Returns a list containing the possible answers (right and wrong), randomly shuffled.
- **validateAnswer(String reply):**Checks if the answer given is correct (comparing with the correct answer, ignoring the upper and lower case).

Description of the functional elements offered through the graphical user interface



The main page, where the user chooses what kind of user he is with the help of a Radio button.



If the user is of the Teacher type , he is greeted with a menu in which he can choose from several functionalities that aim to manipulate the quiz. The choice is made through a radio button.

Quiz Java - Întrebări

Întrebarea 1: Ce reprezintă JVM în Java?

Răspuns corect: Java Virtual Machine;
Răspunsuri greșite: Java Version Manager; Java Virtual Method; Java Very Much;

Întrebarea 2: Care este cuvântul cheie folosit pentru a crea un obiect în Java?

Răspuns corect: new;
Răspunsuri greșite: object; create; instance;

Întrebarea 3: Care este metoda principală într-o aplicație Java?

Răspuns corect: main;
Răspunsuri greșite: start; run; execute;

Întrebarea 4: Care dintre acestea sunt tipuri de date primare în Java?

Răspunsuri corecte: int; float; char;
Răspunsuri greșite: String; ArrayList; HashMap;

Întrebarea 5: Care dintre acestea sunt structuri de control în Java?

Răspunsuri corecte: if; for; while;
Răspunsuri greșite: int; class; public;

Întrebarea 6: Care este pachetul implicit în toate clasele Java?

Răspuns corect: java.lang;
Răspunsuri greșite: java.util; java.io; java.net;

Întrebarea 7: Cum sunt apelate metodele într-o clasă statică?

Răspuns corect: Cu numele clasei;
Răspunsuri greșite: Cu un obiect; Implicit; Cu super;

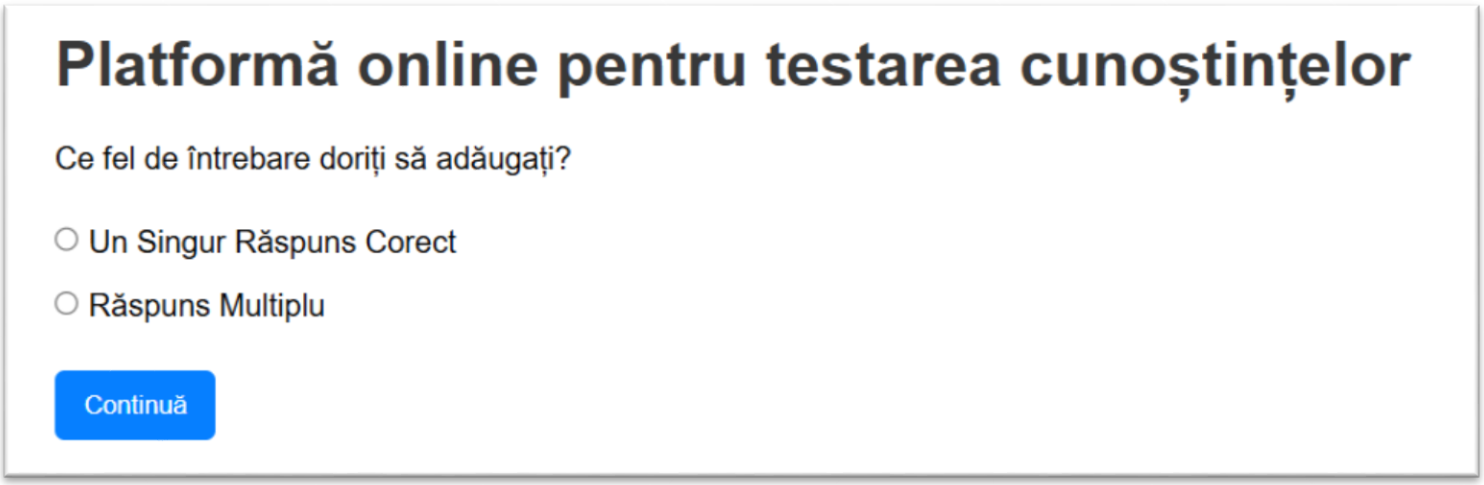
Întrebarea 8: Care dintre acestea sunt metode de colectare în Java?

Răspunsuri corecte: ArrayList; HashMap; Set;
Răspunsuri greșite: BufferedReader; Thread; Runnable;

Întrebarea 9: Care este cuvântul cheie utilizat pentru a moșteni o clasă?

Răspuns corect: extends;
Răspunsuri greșite: inherits; implements; super;

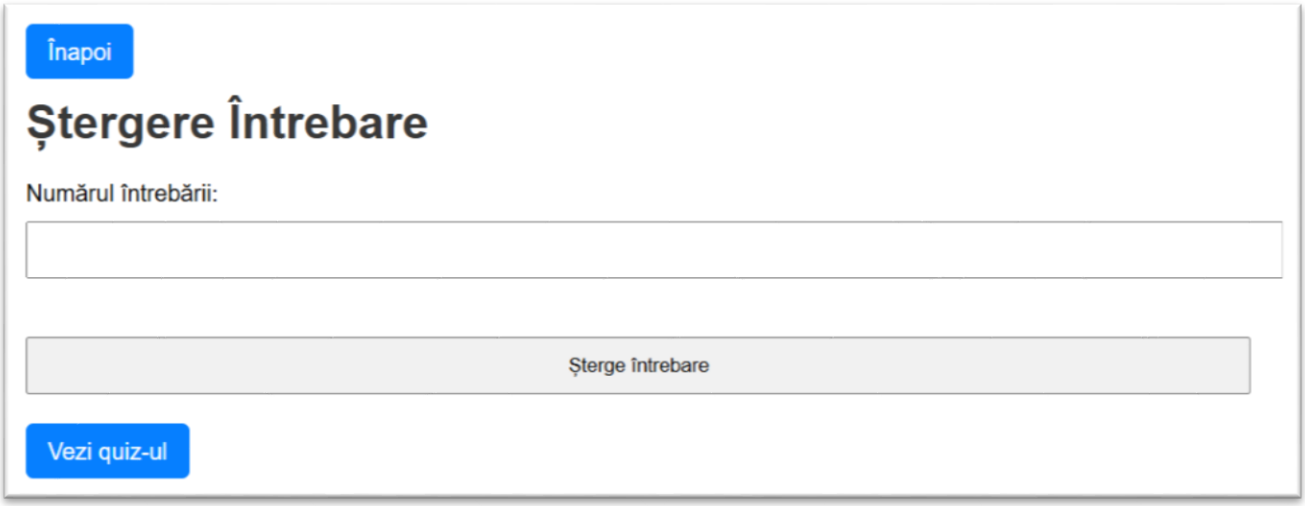
View *Quiz Result*



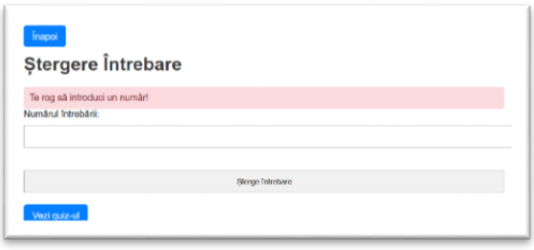
After the Add a Question *feature is called*, the user is asked the type of question they want to add to the quiz

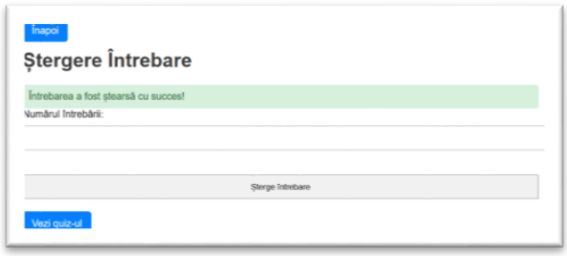
Forms for adding questions

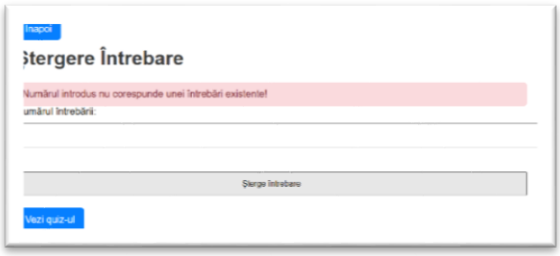
Form for modifying a question



Form for deleting a question

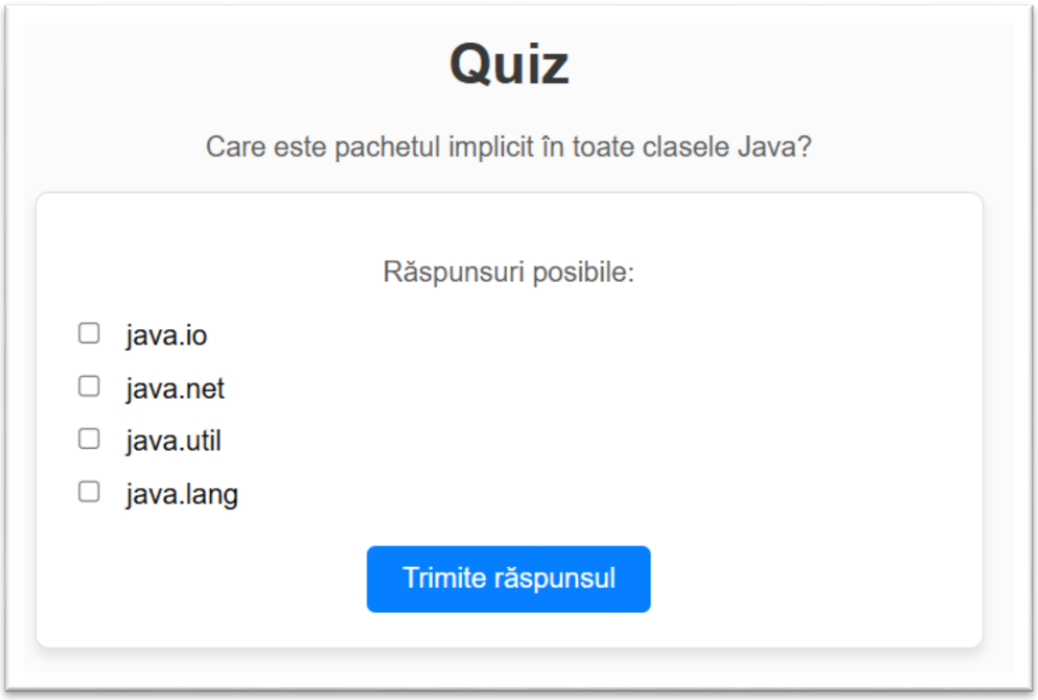




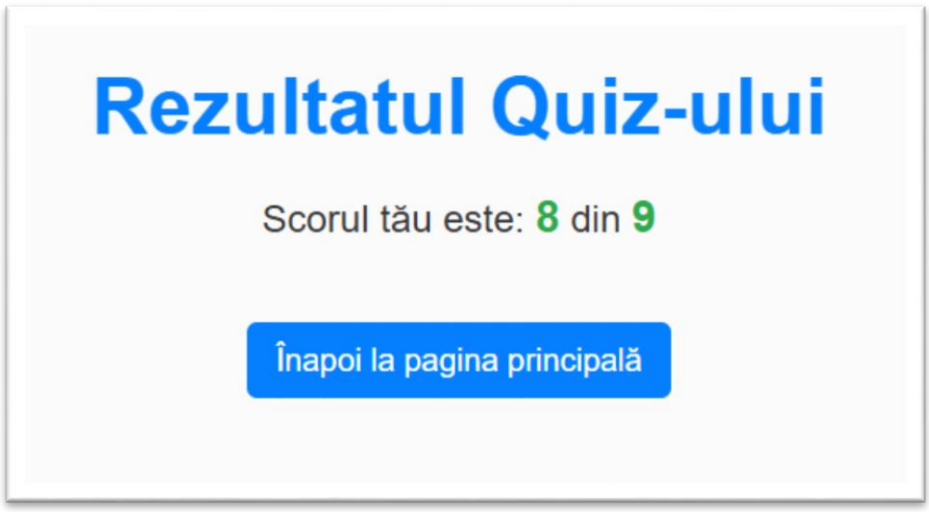


For these forms the user can receive messages

About feature status



If the user is a *Student,* he starts the quiz where he is presented with the questions one by one. The order of the questions and answers is random. The user does not know if a question has a single answer or multiple answer.



After they finish going through all the questions. The user is informed of the result he has obtained.

5 ideas on how the app can be improved

- ***Authentication system and user profiles*** The introduction of an authentication system to allow users to create accounts with the possibility of viewing personalized statistics for each user, such as previous scores, average time per question or percentage of correct answers.

- ***Real-time feedback*** Displaying feedback immediately after each response, indicating whether the response was correct or wrong and providing detailed explanations if applicable. This would improve the learning process of the users.

- ***Adding other question types*** To diversify the content and increase the attractiveness of the app, the following question types can be entered, such as fill-in-the-blank questions and true/false questions.

- ***Introduction*** of a timer system for each question, which would add a difficulty component and a possibility of time-based assessment.

- ***Detailed Post-Quiz Report*** After completing the quiz, users might receive a report that includes the questions they answered wrong, the correct answers