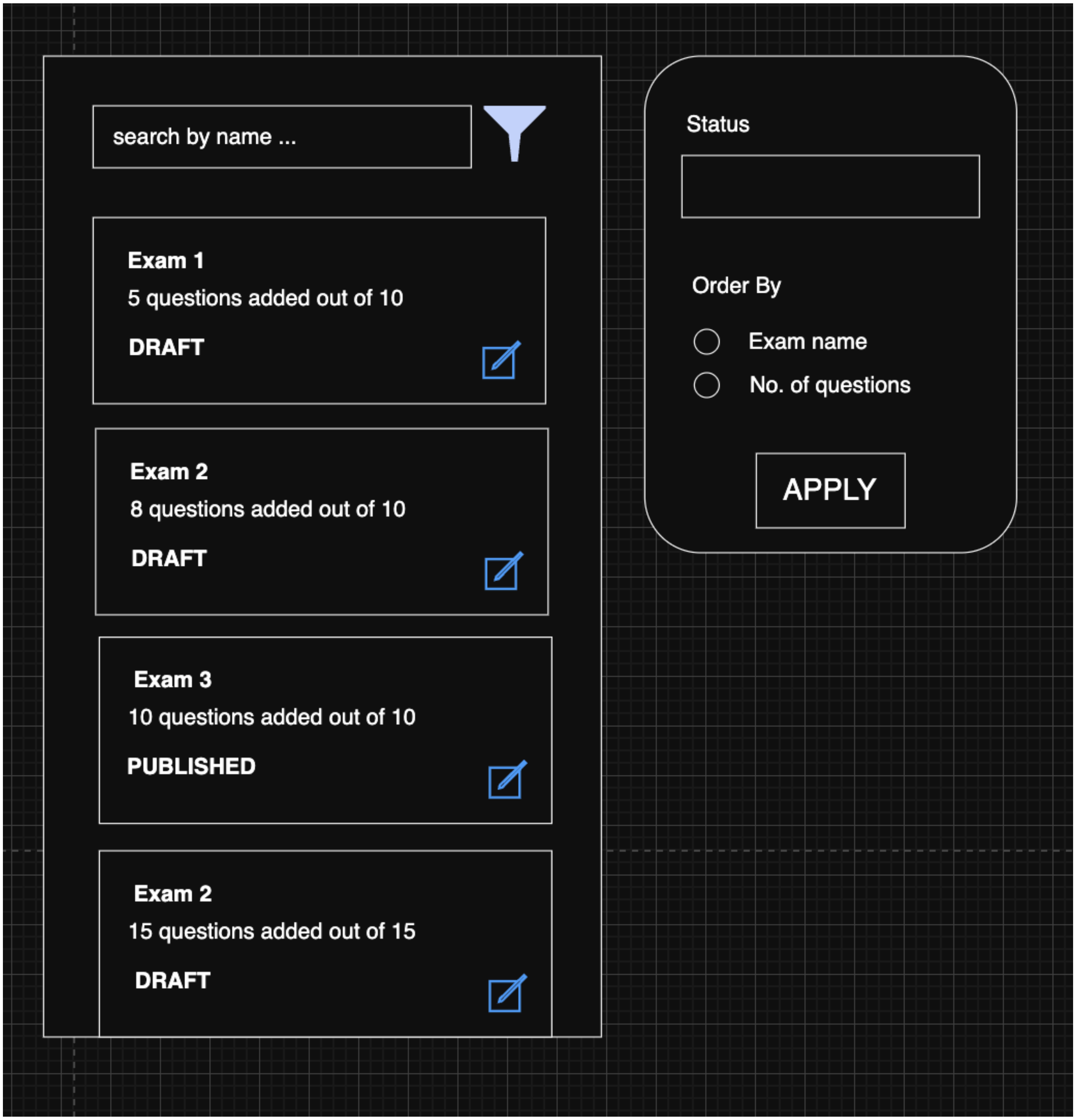
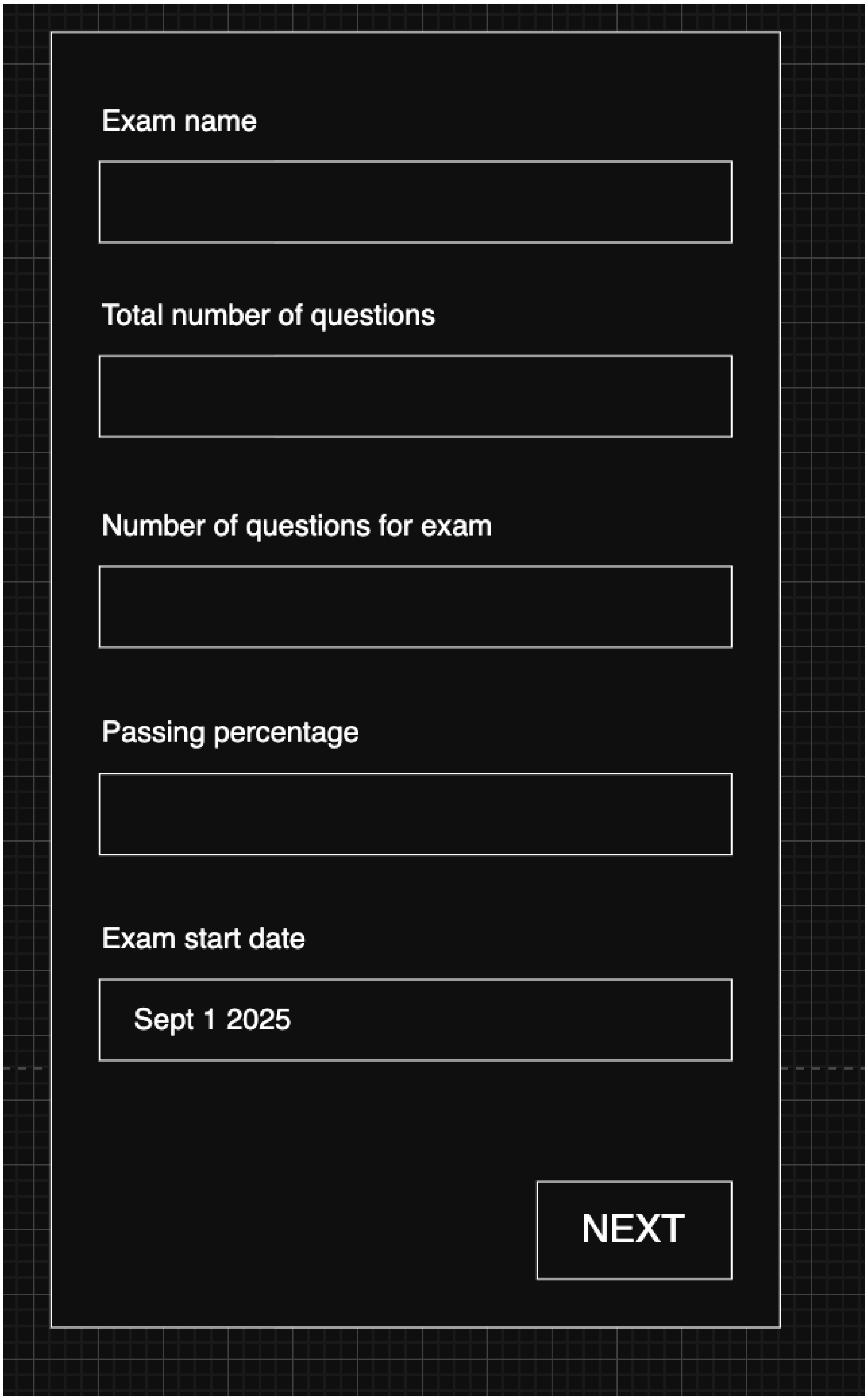


Exam 1
5 questions added out of 10
DRAFT

Exam 2
8 questions added out of 10
DRAFT

Exam 3
10 questions added out of 10
PUBLISHED

Exam 2
15 questions added out of 15
DRAFT



Exam name

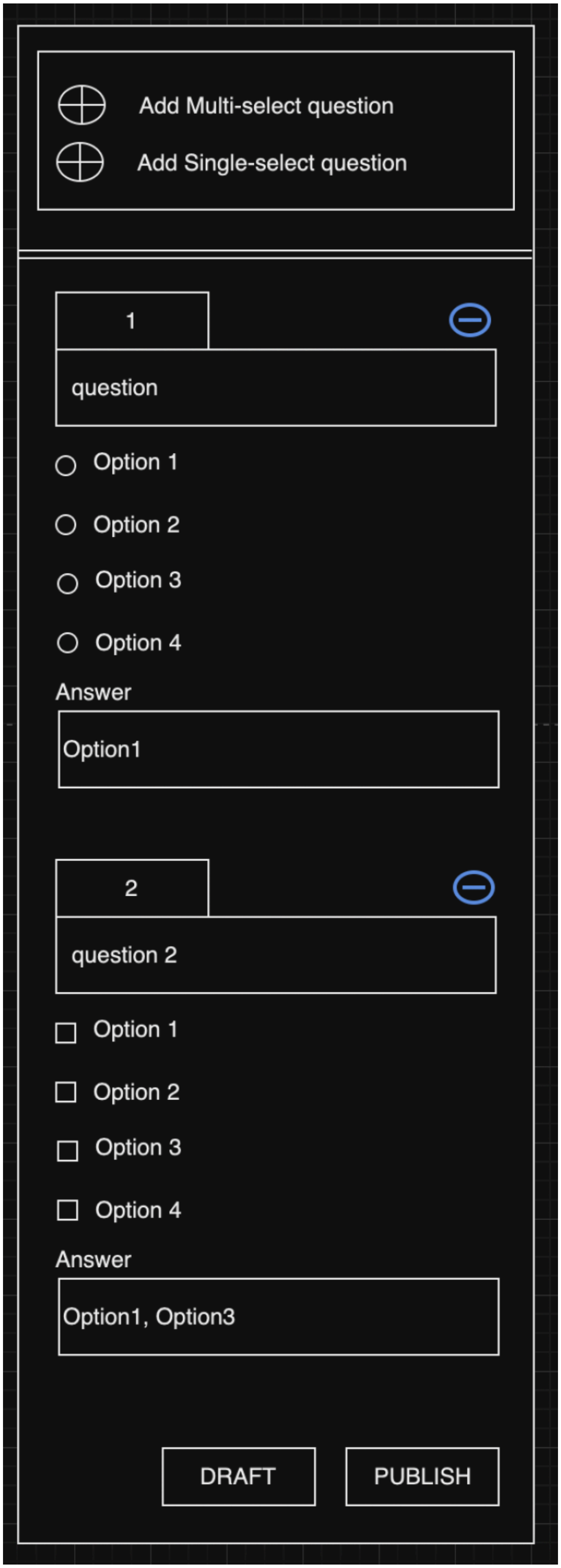
Total number of questions

Number of questions for exam

Passing percentage

Exam start date

Sept 1 2025



1

question

○ Option 1

○ Option 2

○ Option 3

○ Option 4

Answer

Option1

2

question 2

□ Option 1

□ Option 2

□ Option 3

□ Option 4

Answer

Option1, Option3

Exam Overview Screen Admin

- API should support pagination, filtering, and sorting

Create Exam & Add Question Screens

1. Fields for Exam Create and Edit:
    - Name of Exam
    - Total No of Questions
    - No of questions in Exam
    - Passing Percentage
    - Exam Start Date

   Field Validations:
    - All Fields are mandatory
    - No of Questions >0
    - Exam Start Date must be future date
2. Fields for Question Configuration:
    - Multi & Single Select Questions
    - Options for Questions
    - Answers for each question

   Field Validations:
    - Total questions created should be greater than No of questions configured for exam
    - Every question should have an answer based on type of question; Multi select should have more than one answers, single select should have only one answer
    - Question number should be unique and numeric

Exam Screen User

- Number of questions configured for the exam, shuffled per user from the pool of total questions.
- Options for each question shuffled.

Exam Submit User

- Submit the users exam data.
- Calculate score for user and show pass/fail.
- Response: {

  score: 50 of 100
  right_questions: 10
  status: PASS/FAIL

  }

ADMIN API’s

1. exam list with filters
2. Exam create/edit with options
3. Save Exam

User API’s

1. Get exam by id (pass user email as header)
2. Submit exam (pass user email as header)

Guidelines to follow:

1. Use of SQL Database is preferred
2. Use of ORM tools/packages is a plus
3. Do not use any AI tool or plugin for code suggestions
4. Make sure to have proper error handling for APIs and field validations
5. Modular well organized code, ready for production deployment (don't use existing boilerplate, write from scratch)