Schema and template for the Version 2

The Tables/Models of the discussed requirements are as follows: -

1. **User Table: -**

```
Table User {
  id integer [primary key, unique, not null]
  name string [not null]
  username string [not null, unique]
  password string [not null]
  email string [not null, unique]
  login bool [note: 'Check if the user is currently logged in', not null, default: false]
  user_type string [note: 'Role of the user: Admin, Student, Teacher, TA, etc.', not null]
  created_at timestamp [not null]
}
```

2. **Course Table: -**

   This table is used for creating and managing the courses and we can align the courses with every user from this table.

```
Table Course {
  id integer [primary key, unique, not null]
   name varchar(255) [not null, note: "Name of the course, e.g., Machine Learning for Beginners"]
  description text [note: "Description of the course"]
  batch text
   is_public boolean [default: false, not null, note: "Indicates whether students can enroll directly"]
  start_date timestamp [not null, note: "Start date of the course"]
  end_date timestamp [not null, note: "End date of the course"]
}
```

3. **SuperPeer Table: -**

   This table is used for creating the Super Peer's (some extra access to them) for a particular Course under a particular professor.

```
Table SuperPeer {
  id integer [primary key, unique, not null]
  user integer [ref: - User.id, note: "Super Peer ID from user table"]
  course integer [ref: <> Course.id, note: "Course ID of the course"]
```

```
  teacher integer [ref: > Teacher.id, note: "User ID of teacher"]
}
```

4. **Student table: -**

This table is used for associating the students with the courses

```
Table Student {
  id integer [primary key, unique, not null]
  user integer [ref: - User.id, note: "User ID of Student"]
  course integer [ref: <> Course.id, note: "Course ID of the course"]
}
```

5. **Teacher table: -**

The table for handling the teacher of all the courses.

```
Table Teacher {
  id integer [primary key, unique, not null]
  user integer [ref: - User.id, note: "User ID of teacher"]
  course integer [ref: <> Course.id, note: "Course ID of the course"]
}
```

6. **TA table: -**

The table for handling the TA of all the courses.

```
Table TAs {
  id integer [primary key, unique, not null]
  user integer [ref: - User.id, note: "User ID of TA"]
  course integer [ref: <> Course.id, note: "Course ID of the course"]
  teacher integer [ref: > Teacher.id, note: "Teacher ID of teacher"]
}
```

7. **ExamType table: -**

The table for handling the Exam type for the current exam(Quiz, Test, Mid Term, etc.).

```
Table ExamType {
  id integer [primary key, unique, not null]
  name varchar(255) [not null, note: "Type of an exam, e.g., Mid term, end term"]
  course integer [ref: > Course.id, note: "Course ID of the course"]
}
```

8. **Exam table: -**

   The table for handling a particular exam based on the teacher and course type of the exam.

```
Table Exam {
  id integer [primary key, unique, not null]
  examType integer [ref: > ExamType.id, note: "Type of an exam"]
  teacher integer [ref: > Teacher.id, note: "User ID of teacher"]
  course integer [ref: > Course.id, note: "Course ID of the course"]
  date timestamp [not null, note: "Date of the Exam"]
  number_of_questions integer [not null]
  max_scores integer [not null]
}
```

9. **PeerEvaluation table: -**

   The table for handling the evaluation assigned to students, this is created for handling the Normalization issues.

```
Table PeerEvaluation {
  id integer [primary key, unique, not null]
  exam integer [ref: > Exam.id, note: "Type of an exam"]
  document integer [ref: > Documents.id, note: "Type of an exam"]
  evaluator integer [ref: > Student.id, note: "Type of an exam"]
  course integer [ref: > Course.id, note: "Course ID of the course"]
  feedback varchar(255) [note: "Array of Feedback for the evaluation"]
  scores integer [note: "Array of Scores for the evaluation"]
}
```

10. **Documents table: -**

    The table is to maintain the documents of the students which can be uploaded by the student and the teacher or TA also.

```
Table Documents {
  id integer [primary key, unique, not null]
  course integer [ref: > Course.id, note: "Course ID of the course"]
  link varchar(255) [note: "URL of document"]
  avg_score integer
}
```

11. Statistics table: -

The table is for handling the statistics of the exams that are conducted under any course by any teacher.

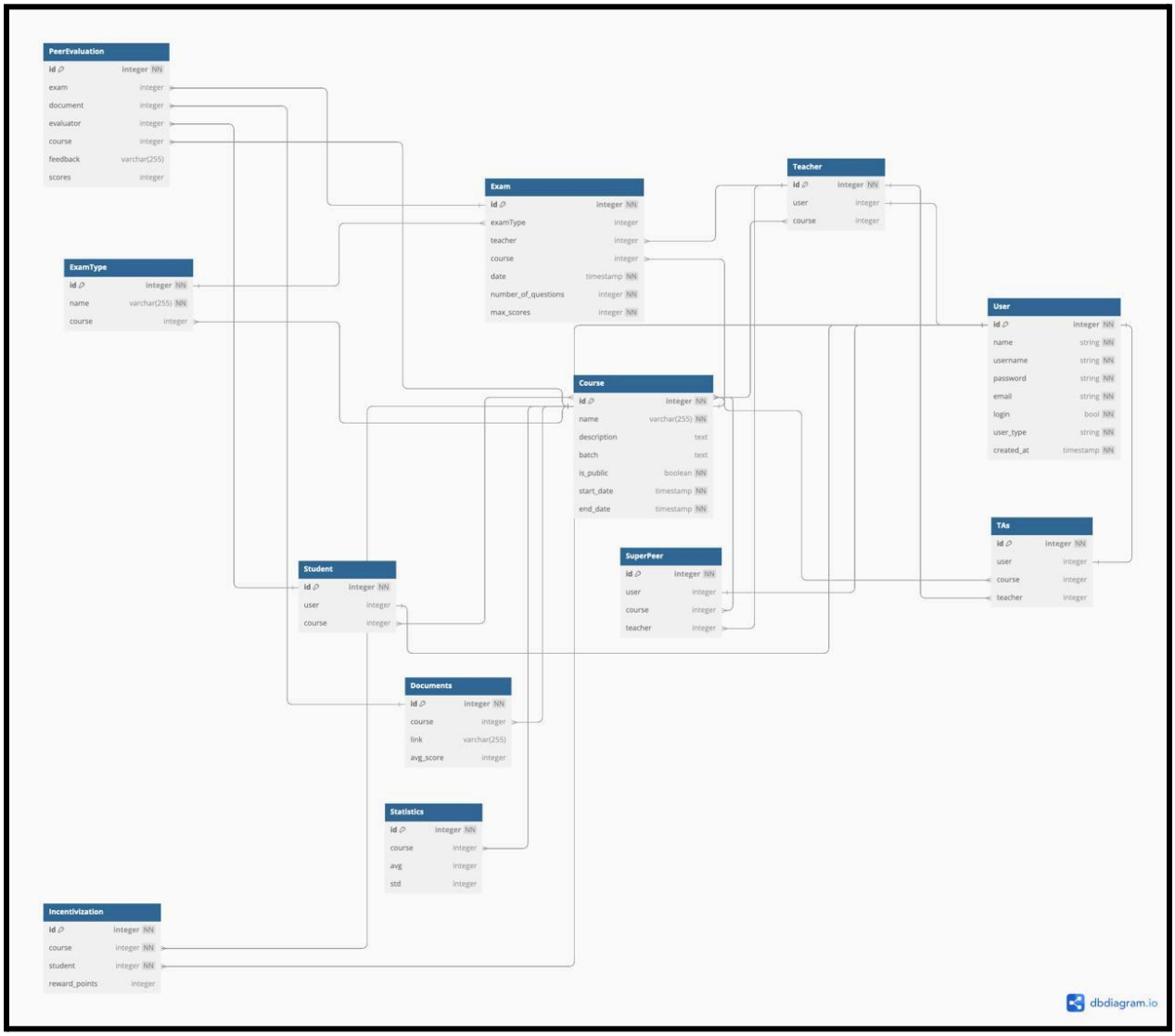
```
Table Statistics {
  id integer [primary key, unique, not null]
  course integer [ref: > Course.id, note: "Course ID of the course"]
  avg integer
  std integer
}
```

12. Incentivization table: -

The table is for handling the incentivization(rewards) for the students based on their evaluation for the exam.

```
Table Incentivization {
  id integer [primary key, unique, not null]
  course integer [ref: > Course.id, not null]
  student integer [ref: > User.id, not null, note: 'User ID of the student']
  reward_points integer
}
```

ER diagram: -


PeerEvaluation
id integer NN
exam integer
document integer
evaluator integer
course integer
feedback varchar(255)
scores integer
Exam
id integer NN
examType integer
teacher integer
course integer
date timestamp NN
number_of_questions integer NN
max_scores integer NN
Teacher
id integer NN
user integer
course integer
ExamType
id integer NN
name varchar(255) NN
course integer
User
id integer NN
name string NN
username string NN
password string NN
email string NN
login bool NN
user_type string NN
created_at timestamp NN
Course
id integer NN
name varchar(255) NN
description text
batch text
is_public boolean NN
start_date timestamp NN
end_date timestamp NN
TAs
id integer NN
user integer
course integer
teacher integer
SuperPeer
id integer NN
user integer
course integer
teacher integer
Student
id integer NN
user integer
course integer
Documents
id integer NN
course integer
link varchar(255)
avg_score integer
Statistics
id integer NN
course integer
avg integer
std integer
Incentivization
id integer NN
course integer NN
student integer NN
reward_points integer
dbdiagram.io