# Real Time match design doc

## Structure:

We have two views:

1. Students' view (Display current question and answer options, at the end display student specific result);
2. Professor's view (Display RT class leaderboard);

## Flow:

1. Professor creates a Match (a WS memory object with a UID, we use the UID in the FE to generate a QR (/join/${match.uid})
2. Students establish WS connections to the match by scanning the QR
3. Professor starts the match manually
4. Students get the next question and possible answers
5. All students answer (or not) until the round specific timer runs out, we go back to 4 until all questions are finished

We will be collecting student results in RT (based on answer speed and correctness, and updating the RT leaderboard in the Professor's view after every question, after match is finished each student on their device gets their specific result ("You placed ${rank} with ${points}");

## Match objects structure:

```
Match {

    students: Map<socketId: string, StudentSession>;

    currentQuestionIndex: number;

    questions: Question[];

}
```

```
Question {
        question: string;
        correctIndex: number;
        answers: string[];
}
StudentSession {
        studentUID: string;
        totalCorrect: number;
        totalPoints: number;
        currentSubmissionIndex: number; // For each new question this resets for each student,
students may select an answer, and change their mind select another before the round timer
runs out
        currentSubmissionTime: number; // The exact time of the submission
}
```

## Point calculation formula:

We calculate points for a correct answer based on the submission time for that answer:

```
points = minPoints + ((Math.min(Math.max(student.currentSubmissionTime - round.startTime,
activeStart), activeEnd) - activeStart) / (activeEnd - activeStart)) * (fullPoints - minPoints);
```

For example, with:

- **activeStart** = 2 seconds
- **activeEnd** = 8 seconds
- **minPoints** = 10
- **fullPoints** = 100
- **roundTime** = 10 seconds

if students answer between [8s, 10s] they get fullPoints,

if students answer between [0s, 2s] they get minPoints,

if they answer between (2s, 8s) they get points based on how soon they answer, example if they answer in the middle point 5s they get 10 + 0.5 * 90 = 55 points, idea is they always get minPoints if they answer correct, and they get additional points based on answer speed

## Match WS events:

### *Server side:*

1. start, ()

// start event is emitted when professor starts the match

2. nextQuestion, (question: Question, startTime: number)

// nextQuestion event is emitted initially and after the timer for the current question finishes, it sends the next question object to the client and the startTime of the question so we can have client-side countdown

3. nextAnswer, (correctAnswerIndex: number)

// after submissions are closed, nextAnswer is emitted with the correct answer, thus showing the students if they answered correct or wrong (red/green on their submitted answer)

4. result, (studentRank: number, studentPoints: number, studentCorrectAnswers: number)

// after all questions, we submit student specific results for all students

### *Client side:*

1. submission, (answerIndex: number)

// whenever the student selects an answer a submission event is emitted to the BE, tracking the submission time of the user, if a user selects another answer a new submission event is emitted with the new answerIndex and a new submission time is registered for that particular student