NUS National University of Singapore

School of Computing

CS3219

Sem 1, AY2023/24

Course Assignments and Course Project Briefing

Download CS3219_Assignments.pdf and CS3219_Project-PeerPrep.pdf from Canvas

---

Bimlesh Wadhwa

Department of Computer Science

bimlesh@nus.edu.sg

Outline

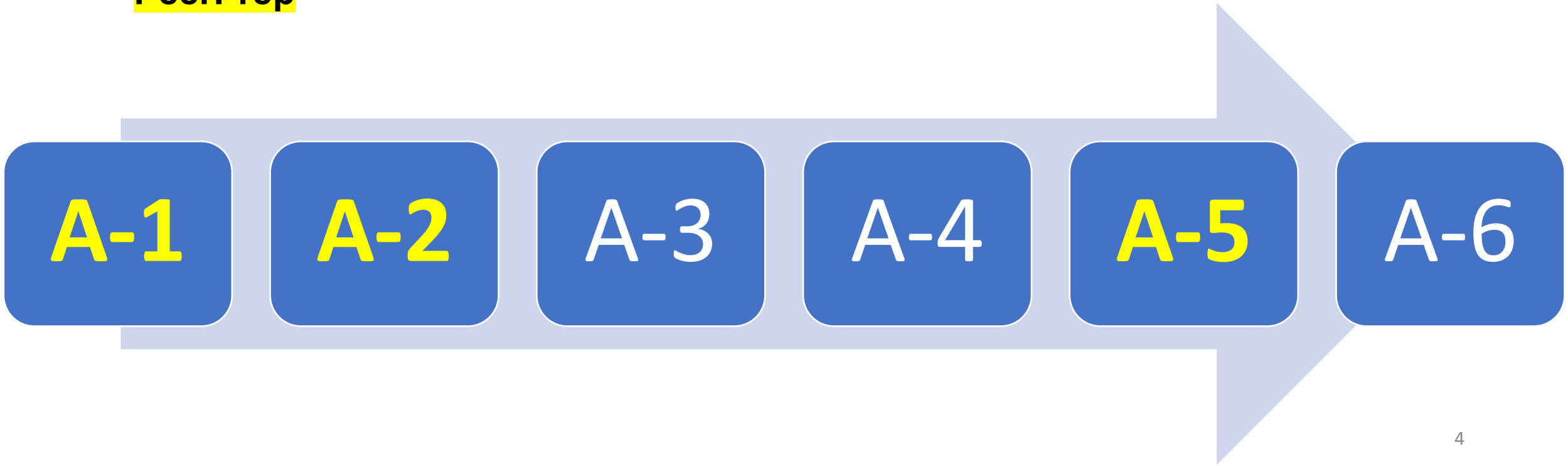
- Assignments
  - Details
  - Work Distribution
  - Submission
- Project
  - Details - Timeline and Deliverables
  - Group Work Distribution
  - Assessment and Feedback
- Support and Communication
- QnA

Assignments

33 POINTS (= 33 MARKS)

Overview

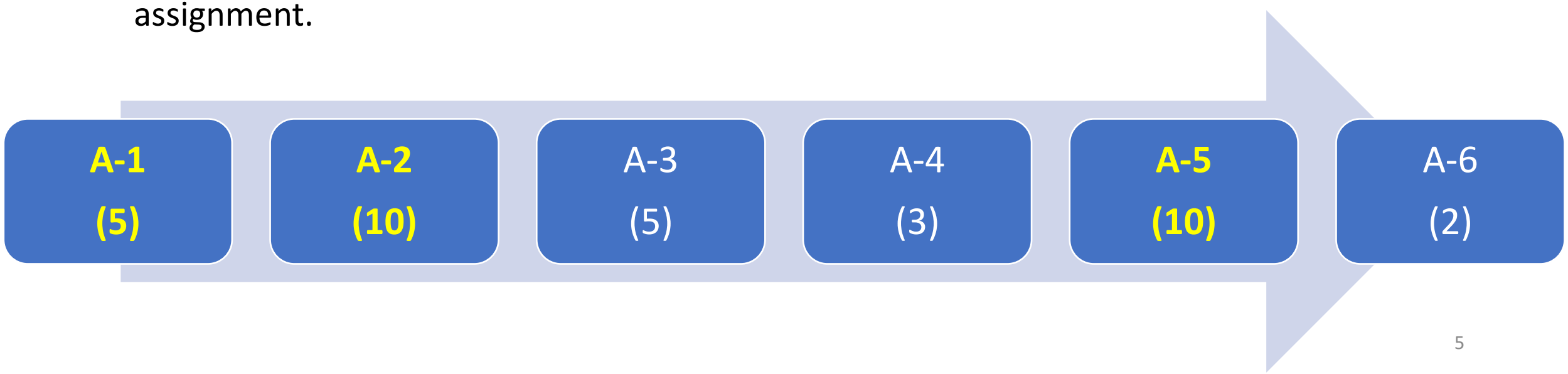
- Scaffolded assignments allow you to learn and use some of the industry-relevant tools and technologies.
- The outcome of these tasks provides you with a base for the course project **PeerPrep**



Overview

- Available - Six assignments totaling 35 points
- Max you can score = 33 points

- If you score full marks on Assignments 1 to 5, you may opt not to do Assignment 6.
- Alternatively, use Assignment 6 marks to compensate for lost marks in any other assignment.



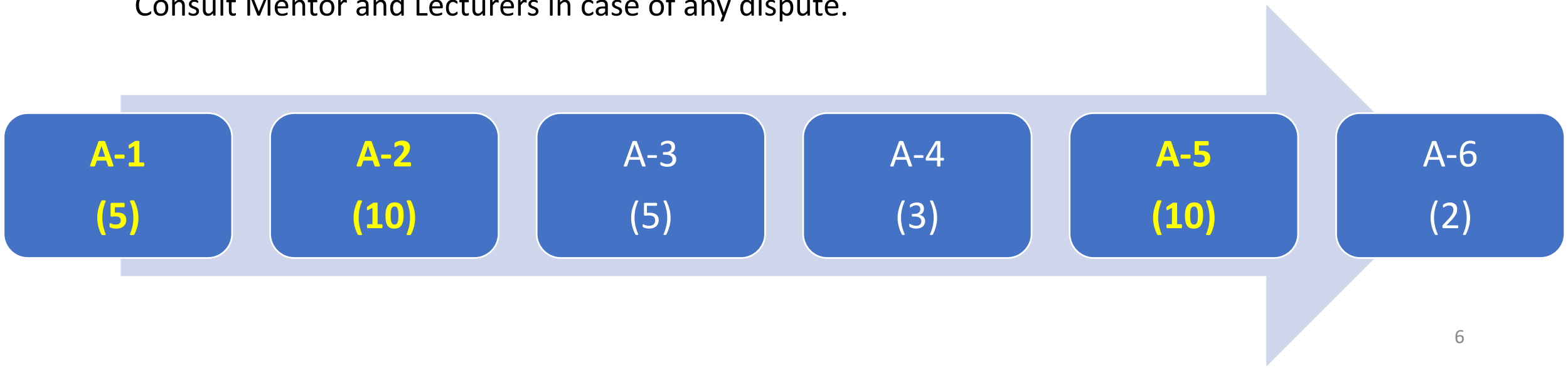
Its all done in your project group

One group submission per assignment.
Project group decides who does what.

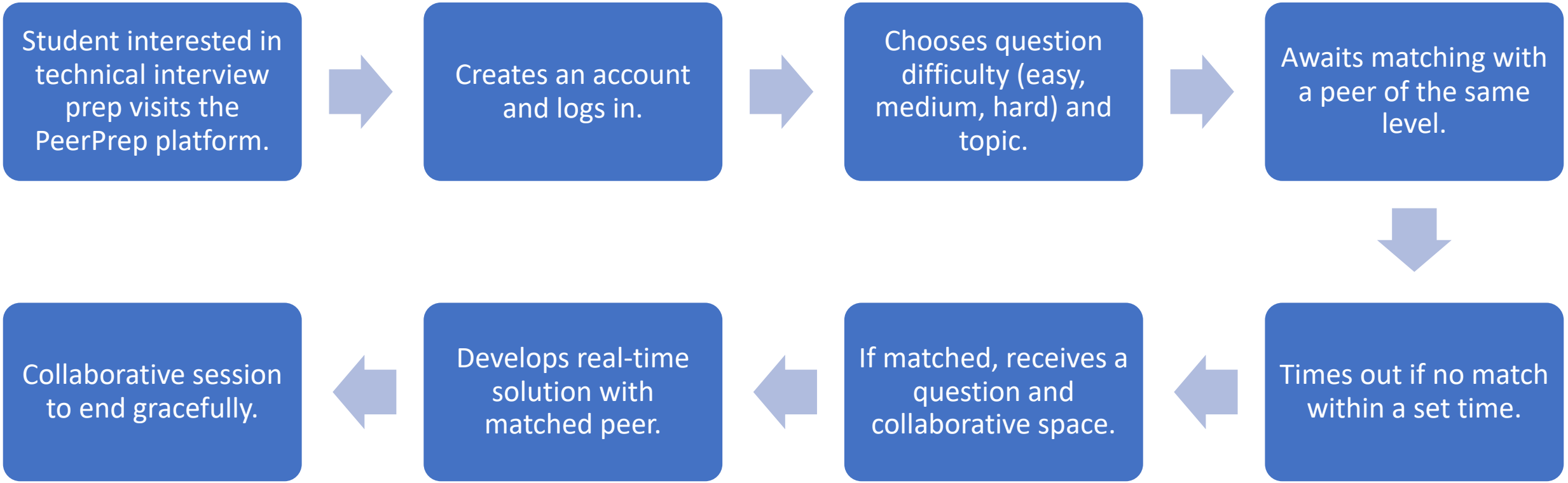
Everyone in the group gets same marks.

Keep organised records of communication and allocation of work.

Consult Mentor and Lecturers in case of any dispute.



A scenario from the PeerPrep Context



Assignment –1 (5 Points)

Develop a SPA to perform CRUD operations on a repository of questions in a question bank.

Assignment -2 (10 points)

To develop functionality for **user profile management**.

To **enhance question repository** created in assignment 1 by making APIs for CRUD operations & maintaining question records in a NoSQL database like MongoDB.

Assignment-3 (5 points)

Implement authentication and authorization for users e.g. to have access to questions, and integrating question repository and user profile management.

Assignment -4 (3 marks)

Containerize all the applications developed from Assignment 1 through Assignment 3.

Assignment – 5 (10 points)

Develop functionality for **matching two users** based on some criteria (like working together on a question of same difficulty level) to facilitate peer preparation for the interview.

Assignment – 6 (2 points)

Write a serverless function to fetch questions from any 3rd party sources and **populate the question repository**.

Work Distribution

Complete these assignments in your respective project groups.

Only one submission per group for each assignment

You may decide to distribute your workload based on skill set and/or interests.

It is expected that each member will equally contribute.

Ensure you do some record-keeping for work allocation and contributions.

Submission

(Upto 20% penalty of max marks on not adhering to instructions)

- Your **code repository** for the assignments will be tracked through GitHub classroom.
  - Follow the submission instructions carefully to correctly tag your repository.

In addition :

- Assignments 1, 2, 3, 4, and 6 **require video submissions** at are no longer than 3 minutes. The contents of the video submission are described in the submission instructions for each assignment.
  - Assignment 5 requires an **in-person demonstration**. Arrange a slot with your project mentor when your assignment is complete.
- **Submission deadline: Wednesday, 15 November 2023 - 5pm.**

Project - PeerPrep

54 POINTS. (27 marks)

Introduction to PeerPrep

- The project involves creating a technical interview preparation platform and peer matching system called PeerPrep, where students can find peers to practice whiteboard-style interview questions together.
- A scenario of the context is as follows:
  - Student interested in technical interview prep visits the platform.
  - Creates an account and logs in.
  - Chooses question difficulty (easy, medium, hard) and topic.
  - Awaits matching with a peer of the same level.
  - Times out if no match within a set time.
  - If matched, receives a question and collaborative space.
  - Develops real-time solution with matched peer.
  - Collaborative session to end gracefully.
- You are free to select the implementation details as long as they align with the main objective of the application.

Choice of Architecture and Technology Stack

- You are free to choose the technology stack to implement PeerPrep.
- Explore and use microservices architecture.
- If your team decides to use any other architecture, please contact your mentor by the **end of week 6** at the latest.

Deliverables: Must-Haves (labelled Mx)

Label	Feature
M1	**User Service** – responsible for user profile management.
M2	**Matching Service** – responsible for matching users based on some criteria (e.g., difficulty level of questions, topics, proficiency level of the users, etc.) This service can potentially be developed by offering multiple matching criteria.
M3	**Question service** – responsible for maintaining a question repository indexed by difficulty level (and any other indexing criteria – e.g., specific topics).
M4	**Collaboration service** – provides the mechanism for real-time collaboration (e.g., concurrent code editing) between the authenticated and matched users in the collaborative space.
M5	**Basic UI for user interaction** – to access the app that you develop.
M6	**Deploying the application** on your local machine (e.g., laptop) using native technology stack. OR Deploy the app on a (local) staging environment (e.g., Docker-based, Docker + Kubernetes).

Here we term them microservices due to our guidance based on microservices architecture.

Deliverables: Nice-to-Haves (Nx)

Label	Feature
N1	Communication: Implement a mechanism to facilitate communication among the participants in the collaborative space (other than the shared workspace). With progressive levels of difficulty: text-based chat service, voice calling service, video (+voice) calling service.
N2	History: Maintain a record of the questions attempted by the user. With progressive levels of difficulty: maintain a list of questions attempted along with the date-time of attempt, maintain a list of questions along with the attempt, maintain a list of questions along with the attempt and retrieve correct answers.

Deliverables: Nice-to-Haves (Nx)

Label	Feature
N3	Code execution: Implement a mechanism to execute code in a sandboxed environment, and retrieve+present the results in the collaborative workspace.
N4	Enhance question service to enable managing questions, for example, tagging (by topic, popularity, etc.,), retrieving questions on the fly during a session initiation.

Deliverables: Nice-to-Haves (Nx)

Label	Feature
N5	Enhance collaboration service by providing an improved code editor. With progressive levels of difficulty: code formatting, syntax highlighting for one language, syntax highlighting for multiple languages.
N6	Enhance collaboration service by providing code translation from one language to another (e.g., JS to Python) depending on the preference of the users. (I.e., user 1 prefers python, user 2 prefers JS, provide a mechanism to translate code to the preferred language of each of the users)

Deliverables: Nice-to-Haves (Nx)

Label	Feature
N7	Incorporate generative AI to assist during the preparation. For example, users can seek help from ChatGPT, Copilot (or such other services) to seek explanation of code written by the other participant. Users can seek help from gen AI to solve the puzzle by providing prompts from within the interface.

Deliverables: Nice-to-Haves (Nx)

Label	Feature
N8	Extensive (and automated) unit, integration, and system testing using CI. Teams can also use various test frameworks or demo CI/CD pipeline
N9	Deployment of the app on the production system (AWS/GCP cloud platform).

Deliverables: Nice-to-Haves (Nx)

Label	Feature
N10	Scalability – the deployed application should demonstrate easy scalability of some form. An example would be using a Kubernetes horizontal pod auto-scaler to scale up the number of application pods when there is a high load.
N11	The application should have an API gateway of some kind that redirects requests to the relevant microservices. An example would be using an ingress controller such as NGINX ingress controller if using Kubernetes (https://kubernetes.GitHub.io/ingress-nginx/).
N12	The application should demonstrate service discovery or implement a service registry of some kind. For example, Kubernetes has built-in DNS features and service networking to pods (https://kubernetes.io/docs/concepts/services-networking/dns-podservice/).

Tasks and Timeline

Week 3: Group formation and release of project document

Week 4: Setup GitHub classroom repository (common repo for assignments and subsequently, project).

Week 5-7: Establish scope of project, develop requirements. Milestone 1 (6/7) - Complete project MVP.

Week 8-10: Continue developing the product, integrate must have microservices. Milestone 2 (week 9/10) - complete developing collaboration service and X nice to have features. Mandatory project update meeting with your mentor.

Weeks 11-13: Complete implementation, integration, testing and deployment. Milestone 3 - submit deliverables - report, code, presentation. Nov 15 Wednesday 5pm

Grading

Total 54 points

- Group- Report [10/54]
- Group- All Mx [10/54]
- Each Sub-Group
  - 3(or more) Nx [24/54]
- Group - Presentation & Demo [10/54]

Please read the project document for details

Support

Project mentor

Teams clarification channel

Teaching Team

SE Toolbox resources

Thank you

Contact me at MS-Teams or

 bimlesh@nus.edu.sg

School of Computing