[CS304] Team Project for 25 Spring

> Total score: 35
>
> Max bonus: +2 (i.e., max total: 37).

A major component of this course is a software development project in team efforts. Your team will propose, design, develop, and maintain a practical and production-level software product.

Project List

The team project should have the students, faculties, and/or staffs of SUSTech as the potential target users. The project's objective should focus on enhancing the efficiency and productivity in users' daily work, study, and/or related activities.

Here are a list of suggested projects.

1. Personal Health Assistant

Maintaining a healthy lifestyle involves regular exercise, proper training, and consistent tracking of physical progress. However, many people struggle with managing their training schedules, staying motivated, and tracking their health statistics. Additionally, the lack of social integration in fitness apps can make it hard for users to stay engaged.

Your goal is to develop a Personal Health Assistant application that provides an integrated platform that not only helps users manage their training routines but also connects them with others for motivation and competition.

Here are some directions you may explore:

- **Training Task Management**: Allows users to create, manage, and schedule their daily or weekly training tasks. Users can set goals, track their progress, and receive reminders for upcoming workouts.
- **Gym Appointment Scheduling**: Enables users to book gym sessions or classes directly through the platform, with availability based on their preferred gym or training center.
- **Training Statistics Dashboard**: Displays statistics on users’ training activities, such as calories burned, workout duration, and progress over time. Users can track fitness goals and see visual representations of their achievements.
- **Online Competitions**: Allows users to participate in online fitness challenges, competing against friends or other users based on specific exercises, performance, or fitness goals.
- **Social Sharing & Billboard**: Users can share their achievements, training milestones, or workout tips with friends, and view a leaderboard showcasing their friends' progress.

2. Smart Photo Album

With the increasing number of photos people capture daily, managing and organizing these images can become overwhelming. Traditional photo management applications offer basic sorting tools but often lack intuitive features for categorizing, searching, and sharing photos.

Your goal is to develop a Photo Album Application that allows users to manage, search, and share photos effectively. The application should provide an easy-to-use, personalized way to interact with photos, organize albums, and share memories with others.

Here are some directions you may explore:

- **Photo Management by Criteria**: Users can organize photos based on various criteria such as location, time, event, or custom tags. Albums can be automatically or manually created to group images that match specific characteristics.
- **Advanced Search Functionality**: Users can search photos by date, location, tags, or even photo content (e.g., facial recognition, objects, etc.).
- **Photo Editing**: Basic in-app photo editing tools like crop and rotate; advanced photo editing tools such as adjustments for brightness and contrast, object removal, and artistic effects.
- **Moment Video & Memory Timeline**: Automatically generates personalized videos by stitching together photos based on selected criteria like an event, trip, or time frame. These videos can be customized with music, transitions, and captions, and users can relive their moments through a personalized, chronological timeline.
- **Sharing & Social**: Users can share photos or albums with friends and family via social media or within the app. Albums can be set to private or public, and collaborative albums can allow multiple users to contribute and view shared photos.

3. Coursework Grading System

CS students use Blackboard (BB) for text and image-based assignments submission, Online Judge (OJ) for code submissions, while midterm and final exams are assessed through traditional paper-based grading.

However, this fragmented approach presents several challenges. Students must navigate multiple platforms, making assignment management cumbersome. Code evaluation can be inconsistent due to variations in test case quality. Handwritten work can affect final scores due to handwriting quality. In addition, high website traffic during deadlines can cause delays, making submissions stressful and unreliable.

Your task is to design and implement an improved Assignment Grading System that addresses these limitations. The system should not be a replica of Blackboard, Sakai, or Online Judge. Instead, it should either introduce a completely new design or significantly enhance existing systems to resolve the identified problems.

Here are some directions you may explore:

- **Unified Platform**: Create a single system that supports text, code, image, and document-based assignments, eliminating the need for multiple submission platforms.
- **Handwritten Work Processing**: Enable students to upload handwritten assignments, which can be automatically converted to PDF/DOC for online viewing and grading.
- **Automated & Assisted Grading**: Implement AI or rule-based tools to assist in grading handwritten coursework and exams, reducing manual effort.
- **Performance Optimization**: Ensure smooth platform access, even during peak submission times, through caching, load balancing, or other performance-enhancing techniques.
- **User Experience Improvements**: Focus on intuitive UI, real-time feedback on submissions, and seamless integration with university systems.

4. Intelligent Course-Aware IDE

Traditional coding environments focus primarily on programming tasks, requiring students to switch between multiple tools for lecture notes, assignments, and discussions. This fragmented approach disrupts the learning process, making it difficult to connect theoretical concepts with practical coding exercises. Additionally, students often struggle with organizing course materials, revisiting key concepts, and efficiently reviewing content.

Your task is to design and implement a course-aware IDE that integrates coding with learning resources. The IDE should allow students to manage course materials, take notes, execute code snippets in slides, take exercises, learn from examples, and track progress within a unified environment. This tool could be a standalone application or a plugin/extension for existing IDEs. The goal is to transform how students engage with programming courses by making lectures more interactive, learning more intuitive, and collaboration seamless.

Here are some directions you may explore:

- **Intuitive Course Resource Management**: Provides a structured interface for organizing course materials. Users can take digital notes directly within the IDE, bookmark key concepts, and link resources to coding exercises for easy reference.
- **Code-Linked Lecture Execution**: Lecture slides are embedded within the IDE, allowing students to modify and execute code snippets inline. This feature enables real-time experimentation with lecture examples and immediate feedback on code correctness.
- **Collaborative Coding & Resource Sharing**: Supports real-time collaborative coding, discussion threads linked to lecture content, resource sharing among students, and built-in chat for team projects.
- **Course Progress Tracking**: Tracks student progress through the course materials, showing completed tasks, upcoming assignments, and contents needing attention.
- **AI Learning Assistant**: Summarizes lecture content, highlights key takeaways, generates mind map or flow charts for visualization, and creates quizzes for self-assessment with auto-grading and explanations.

5. SUSTech Car Racing Game

Every fall, SUSTech welcomes new students, and freshmen often need some time to become familiar with the campus and their new environment. Navigating through the vast campus can be a challenge, especially when adjusting to new schedules and locations.

To help freshmen adapt to SUSTech life, you are tasked with creating a SUSTech Car Racing Game to help students explore the campus in a fun and engaging way. By racing through a virtual map of SUSTech, players can become familiar with key landmarks and locations such as the library, student center, and academic buildings, all while enjoying the excitement of a car racing game.

Here are some directions you may explore:

- **SUSTech Campus Map**: The game will feature a detailed, interactive map of SUSTech, allowing players to race through key landmarks and locations, such as the library, student center, and academic buildings.
- **Track Obstacles and Shortcuts**: The SUSTech campus map includes unique track obstacles, such as traffic cones, benches, and construction areas, as well as hidden shortcuts that players can discover to gain an edge over opponents.

- **Themed Vehicles & Power-Ups**: Players can choose vehicles inspired by different SUSTech departments and colleges (e.g., a "CSE Turbo" car). Unique power-ups, such as a "Physics Gravity Boost" for speed bursts or a "Mathematics Barrier" for temporary invincibility, add strategic depth to the races.
- **Diverse Racing Modes**: Players can compete in time trials, AI races, or multiplayer races against friends and other players online.
- **Dynamic Weather and Day/Night Cycle**: The game simulates different weather conditions and a day/night cycle that affects visibility and track difficulty.

Bonus by Project

The bonus points varies by project:

- **Personal Health Assistant** – No Bonus
- **Photo Album Application** – No Bonus
- **Coursework Grading System** – ⭑ +1 Bonus
- **Intelligent Course-Aware IDE** – ⭑⭑ +2 Bonus
- **SUSTech Car Racing Game** – ⭑⭑ +2 Bonus

To quantify for the bonus, your project must reach a sufficient level of maturity. **A half-finished or poorly implemented project will not be eligible for bonus**.

Basic Requirements

1. The final software product should be functional, operational, and user-friendly, providing a meaningful solution to the problem at hand. Additionally, your project should include at least two major iterations (sprints), and each sprint should deliver executable and demonstrable features or improvements. Detailed expectations for each sprint will be provided in the milestone descriptions.
2. Your project should implement **at least 5 distinct and notable features**. These features should be clearly distinguishable from one another and add significant value to the project, which will be outlined later in the project proposal requirement. The features listed in this document are only examples but not hard requirements; you can propose other features based on your project's needs and goals.
3. The project can take the form of a standalone desktop application, web application, mobile app, toolkit, or another suitable software product that satisfies the requirements outlined. Most projects will be implemented using Java, Python, or C++, and will run on Unix, Windows, or Macintosh operating systems. However, you are welcome to choose the technology stack that best fits the specific needs and goals of your project.
4. All projects must use `git` in GitHub Classrooms` to manage the source code, documentation, and other essential project artifacts. This ensures proper collaboration, code history tracking, and project management. See lab 1 for details.

Teams

Please form a team based on the following requirements:

- The size of the team must be 4 or 5. A team of size less than 4 or larger than 5 is NOT accepted.
- Teams will make a series of project presentation during labs throughout the semester. **EVERY team member needs to show up during the presentation**. For this reason, we recommend you to find team members from the same lab session.

- If you have to team up with students from other lab sessions, you should choose which lab session will your team be presenting throughout the semester, and this lab session **could not be changed later**.
- The presentation must be conducted in the lab session enrolled by **the majority of the group members**. For example, in a 5-member group, if 3 members are in Tao's Tuesday 3-4 lab session and 2 members are in Zhao's Tuesday 3-4 lab session, the presentation must be held in Tao's Tuesday 3-4 lab session.

Please see our lab 1github-classroom tutorial for how to create a team in GitHub Classroom.

Project Management

In each team, one member will take on the role of Project Manager (PM), which combines the responsibilities of both the Product Owner and Scrum Master in a Scrum framework.

The PM is expected to keep track of each member's contributions, making sure that the workload is fairly distributed and that any performance issues are addressed promptly. This includes coordinating tasks, resolving conflicts, and helping to maintain effective communication among team members. Students in the PM role will **earn a 3% bonus of the team's project score** for their efforts in managing the project effectively.

If no PM is selected, there will be no penalty. However, every team member will receive the same project score **without any contribution adjustment**, and the team is not eligible for the kick-out policy, see below.

Contributions and "Kick-out" Policy

As PM ensures that all team members contribute equally to the project, **every team member will receive the same grade for the project**.

If a team member consistently underperforms and negatively impacts the team's progress, the PM and the team may decide to **kick out one underperforming member at Milestone 2 (Week 9)**. In order for a team member to be removed, the PM must have clear evidence to support this decision, such as documented instances of missed deadlines and lack of contribution.

After kicking out a member, the team is not required to recruit a new member, even if the team size is reduced to 3. However, if the team chooses to recruit a new member, that is also acceptable.

Once being removed, the underperforming member has the option to either:

- Find another team to join for Milestone 3, or
- Go solo for Milestone 3.

The scores for Milestones 1 and 2 will remain unchanged for the removed team member, regardless of whether they join another team or work independently.

The goal of this policy is to ensure fairness and accountability within the team, and to encourage every member to actively contribute to the project's success.

Milestones & Deliverables

The team project has 3 milestones, in week 5 (proposal), week 9 (the 1st sprint), and week 15/16 (the 2nd sprint), respectively.

At each milestone, **each team** is required to:

- **Submit a set of deliverables**. Typical deliverables include working code, documentation, report, configuration files, test suites, etc.
- **Deliver a 10-minutes presentation** during the lab session.

The three primary criteria for a successful project are: satisfying the client's needs, usability of the product, and maintainability over the life of the product. Please take these criteria in mind when developing your project.

Detailed grading scheme will be released before each milestone.

Code Reuse & AI Usage Policy

1. You MUST NOT directly reuse an entire existing open-source project and claim it to be your team project.
2. You MUST NOT reuse or adapt existing projects from other courses (e.g., OOAD, Innovation Practices), competitions, or from previous semesters of CS304.
3. You may reuse third-party libraries, frameworks, and APIs as part of your project. However, the core functionalities of the product must be implemented by your team members. You are also required to provide proper references for any third-party tools or resources you use.
4. As we will explore throughout the course, AI is transforming the software engineering landscape. You are encouraged to leverage AI tools throughout the development process of your team project. However, any code, scripts, configurations, documentation, or other artifacts generated or influenced by AI **must be accompanied by clear and concise references**. You must specify the AI tools, versions, models, prompts, usage methods, and any adaptations made to the AI-generated content. Detailed instructions for how to properly reference AI tools and usage will be provided in the description of each milestone.

Any violation of the above policies will be considered a breach of academic integrity and will result in serious penalties, as outlined in the *Regulations on Academic Misconduct in Courses for Undergraduate Students*.