CSCI3100 Tutorial 1: Project Introduction

January 16, 2023

JIANG, Shuyao

Objective

- Practice what you are learning in this CSCI3100 Software Engineering course by **designing**, **implementing**, **testing**, and **documenting** a typical software engineering project (e.g., a **web-based client-server application**, or a **software game application**).

Modern Application

airbnb

Gmail

amazon

Booking.com

Uber

YAHOO!

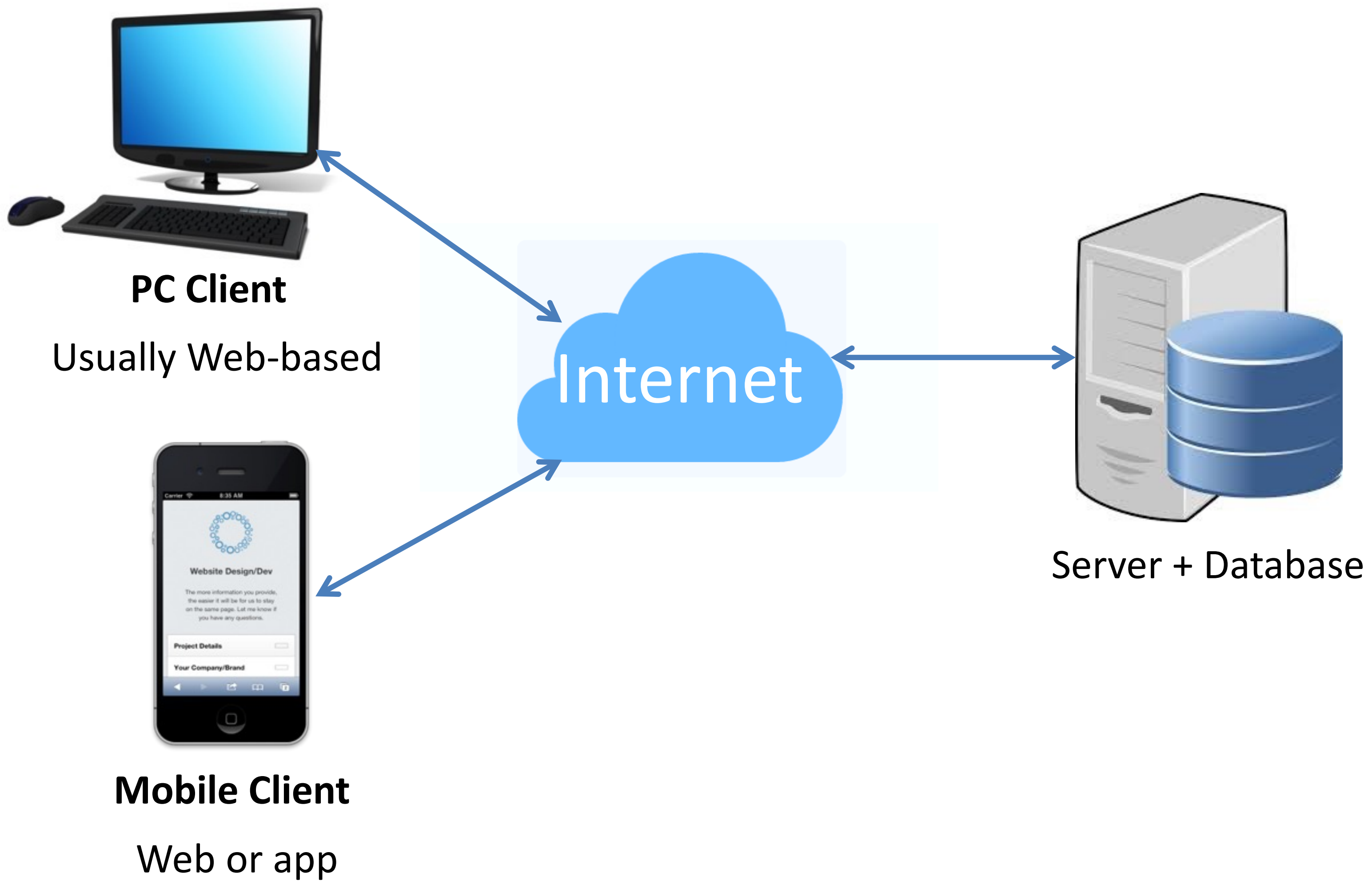
in

香港天文台
HONG KONG OBSERVATORY

The Government of the Hong Kong Special Administrative Region

恒 生 銀 行
HANG SENG BANK

Common Architecture



Important Stats

- Project accounts for 40% of the course grade.
- There are 5 phases in the project.

Phase Deliverables	Weightings
1. High-Level Design Document	5%
2. DFD Specification Document and GitHub Repository Creation	10%
3. UML Specification and UI Design Document	15%
4. Project Demo	60%
5. Testing Document and Final Commented Code	10%

Project Topic

- We provide **4** applications for your selection.
  - *Simplified Twitter*, *course selection system*, *Pac-Man*, and *Gobang*
- We define "**Basic Requirements**" and "**Advanced Functionality Suggestions**" for your project.
  - Basic Requirements (70%): Features that your project must have.
  - Advanced Functionality Suggestions (30%): Features that are optional for your reference.
- The detailed application description and requirements can be found in **Appendix 1**.

Topic 1: Twitter

- Twitter is a microblogging and social networking service. Users can post and interact with messages (“tweets”).
- **Basic Requirements**
  - Client-server architecture
  - Global Database
  - User Interface
  - User Management
  - Admin User
  - User Operations
    - Search for users, Follow other users, Like/dislike a tweet, Comment a tweet, Retweet a tweet, Post a tweet, Show other users’ tweets

Topic 1: Twitter

- **Advanced Functionality Suggestions**
  - Pretty UI
  - Privacy Control
  - User Recommendation
  - Tweet Recommendation
  - Private Chat
  - Video tweets
  - …

Topic 2: Course Selection System

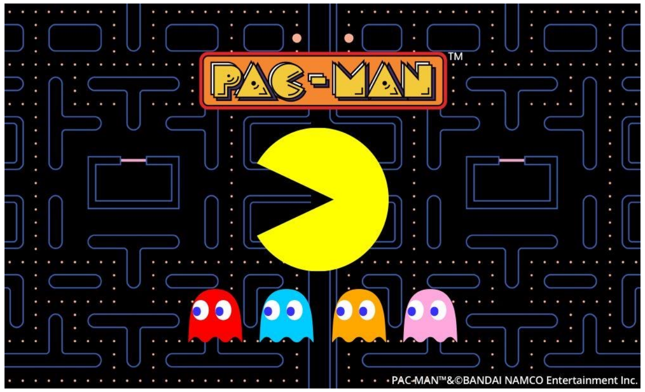
- **Basic Requirements**
  - Client-server architecture
  - Global Database
  - User Interface
    - A course browsing page and a profile page
  - User Management
  - Admin User
  - User Operations
    - Search for courses, Select courses, Show selected courses, Drop courses

Topic 2: Course Selection System

- **Advanced Functionality Suggestions**
  - Pretty UI
  - Concurrency control
  - Course outline upload (admin side) and view (user side)
  - Schedule display
  - …

Topic 3: Pac-Man

- Pac-Man is an interactive computer game developed in the early 1980s. It was one of the most popular games at that time. It is still being played by many people.
- **Basic Requirements**
  - User Interface
    - Menu items, Title screen, Characters, Maze and pac-dots, Messages
  - User Management
  - Database
  - Functional Requirements
    - Basic gameplay, Character behaviors, Scores, Completing a level, Game over

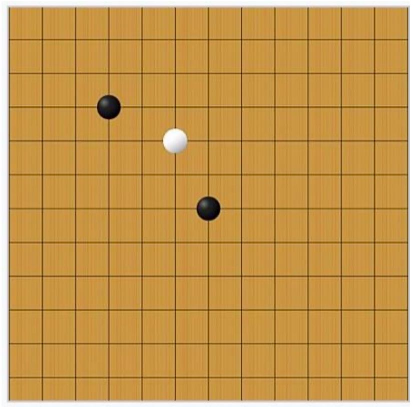


Topic 3: Pac-Man

- **Advanced Functionality Suggestions**
  - More complicated game logic
  - More complicated designs of levels
  - Machine-controlled mode
  - 3D game
  - Use of sound in the game
  - Prettier UI
  - …

Topic 4: Gobang

- Gobang is a classic strategy board game on a Go board. Players alternate turns placing a stone of their color on an empty intersection.
- The side that first forms five consecutive pieces of the same color on the horizontal, vertical, and diagonal directions of the board is the winner.



Topic 4: Gobang

- **Basic Requirements**
  - Components
    - A 19×19 Goboard, 2 players, different stones for different players
  - Player Type
    - 2 human players / 1 human player + 1 random player
  - User Management
  - General Game Logics
    - Player move, Game over, Retract a false move
    - Show the information: Start time, Elapsed time, All player names, Current player and its stone type, Current Goboard with stones

Topic 4: Gobang

- **Advanced Functionality Suggestions**
  - Support (some of) the functionalities in more complicated game control
    - Score system, Chit-chat during game, Add friends and invite friends to a game, Early Termination of the Game
  - detect and disallow (some of) the forbidden moves
  - Implement Game AI
  - Sound effects of the game
  - Pretty UI
  - …

Phase 0: Forming Project Team

- **4-5 students** for each group.
- All students in a group work on the same project for the entire project duration.
- **No joint work** over any technical aspects of the project is allowed between any two teams.
- Deadline: **Jan 20 (Fri.)**
- Link: https://forms.gle/dmtQ9RyUYSQ8jETRA

Phase 1: High-Level Design

- **Duration:** 2 Weeks (**Feb 4 23:59:59pm**)
- **Grade Weighting:** 5%
- Submit a **high-level design document** to provide high-level descriptions of functionalities, features, and architecture design of your application.
- **Feedbacks** will be provided on your high-level design. You should **reconsider** and possibly **revise** the project goals.

Phase 2: DFD Specification and GitHub Repository Creation

- **Duration:** 3 weeks (**Feb 25 23:59:59pm**)
- **Weighting:** 10%
- You need to complete two tasks:
  - Specify your application functionalities with data flow diagrams (DFDs).
  - You will work as programmers to implement your own design and collaborate using the **git** version control system.
- At the end of this phase, you are required to
  - Submit the **DFD Specification Document**.
  - Create a **code repository on GitHub**.

Phase 3: UML Specification and UI Design

- **Duration:** 4 weeks (**Mar 25 23:59:59pm**)
- **Weighting:** 15%
- You are expected to use **UML diagrams** to specify your application and refine your UML diagrams during your implementation.
- You should also describe the **user interfaces** of your application.
- You are required to submit a **UML Specification and UI Design Document** by the end of this phase.

Phase 4: Project Demo

- **Duration:** 2.5 weeks
  - Demo Day: **Apr 13 & 14 (two days)**
- **Weighting:** 60%
- In this phase, you are completing your project. You will need to make a **demonstration** of your complete application (15 minutes per group).
- Signup schedule for demonstration will be announced on the **course website**.

Phase 5: Testing and Final Commented Code

- **Duration:** 3 weeks (**May 6 23:59:59pm**)
- **Weighting:** 10%
- You are expected to conduct testing on your application. You should describe the **test plan** in your testing document.
- Your **final code** is also required. The code should also be **commented** as detailed as possible. A **README** should be included.
- You are required to submit your **testing document and final code**.

Grading Criteria

- **Documents:** Based upon the **technical content** and the **clarity of the presentation**.
  - Please refer to **Appendix 2** for more information.
- **Demo:** Based upon the **functionalities** of your application.
  - Basic Requirements: 70% (Refer to **Appendix 4**)
  - Advanced Functionality Suggestions: 30%
- **Final code:** Based upon the availability of the **README** and the **readability** of your code.
- The overall **quality** and **functionality** of the project is the key scaling factor for all aspects.

Grading Criteria

- Project grade will be based for the **whole team** and will **NOT** be assigned **individually** to members.
- However, complaints about **free-riders** will be considered during project development and will be verified in Demo Day.

Submission (Report)

- Each project group should submit the softcopy of the report and the **VeriGuide** recipient to **Blackboard** before the deadlines.
- File names (**Important!**) :
  - "Group** High-Level Design Document"
  - "Group** High-Level Design Document VeriGuide"
  - "Group** DFD Specification Document"
  - "Group** DFD Specification Document VeriGuide"
  - "Group** UML Specification and UI Design Document"
  - "Group** UML Specification and UI Design Document VeriGuide"
  - "Group** Testing Document"
  - "Group** Testing Document VeriGuide"

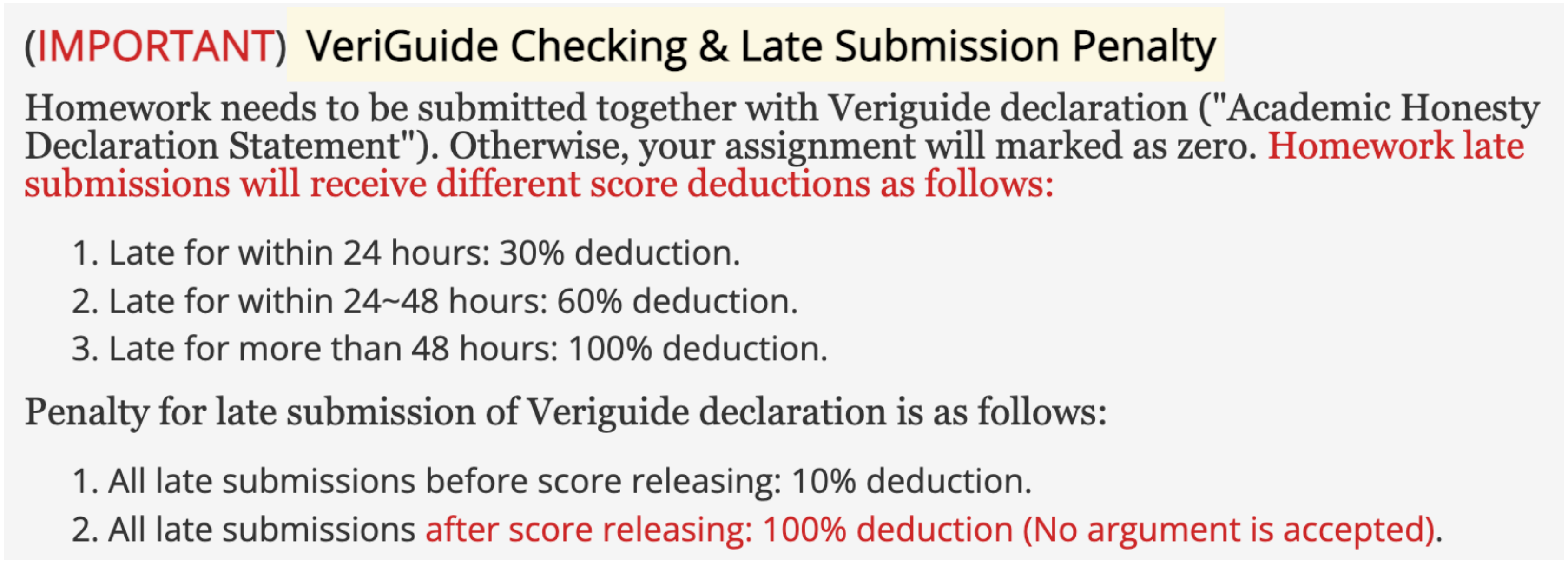
(replace ** with your **group ID**) (without quotes)

Submission (Code)

- **ALL** project stuff (source code, images, databases files, etc.) should be maintained with **Git**.
- You **MUST** submit your project to GitHub and **faithfully** record your **coding activities**.
- We will **NOT** accept submissions via other approaches.
- Tutors will **NOT** help you debug your code.
- A detailed guide for code submission is in **Appendix 3.**

Late Submission Policies

- The late submission and missing Veriguide receipt follow **the same policy as assignments**. You can find the policy on the course website.



(IMPORTANT) VeriGuide Checking & Late Submission Penalty

Homework needs to be submitted together with Veriguide declaration ("Academic Honesty Declaration Statement"). Otherwise, your assignment will marked as zero. Homework late submissions will receive different score deductions as follows:

1. Late for within 24 hours: 30% deduction.
2. Late for within 24~48 hours: 60% deduction.
3. Late for more than 48 hours: 100% deduction.

Penalty for late submission of Veriguide declaration is as follows:

1. All late submissions before score releasing: 10% deduction.
2. All late submissions after score releasing: 100% deduction (No argument is accepted).

Requirement: Technical

- Frontend: Web based access.

  The server-side program is recommended to be built on Node.js. PHP, or Django is also acceptable.

Why node js ?

①High-Performance
②Easy to modify and maintain

Requirement: Technical

- Backend: Database.

  SQL database (e.g., MySQL, or Sqlite), or NoSQL database (e.g., MongoDB, or Redis) MUST be employed for storing data.

Requirement: Programming

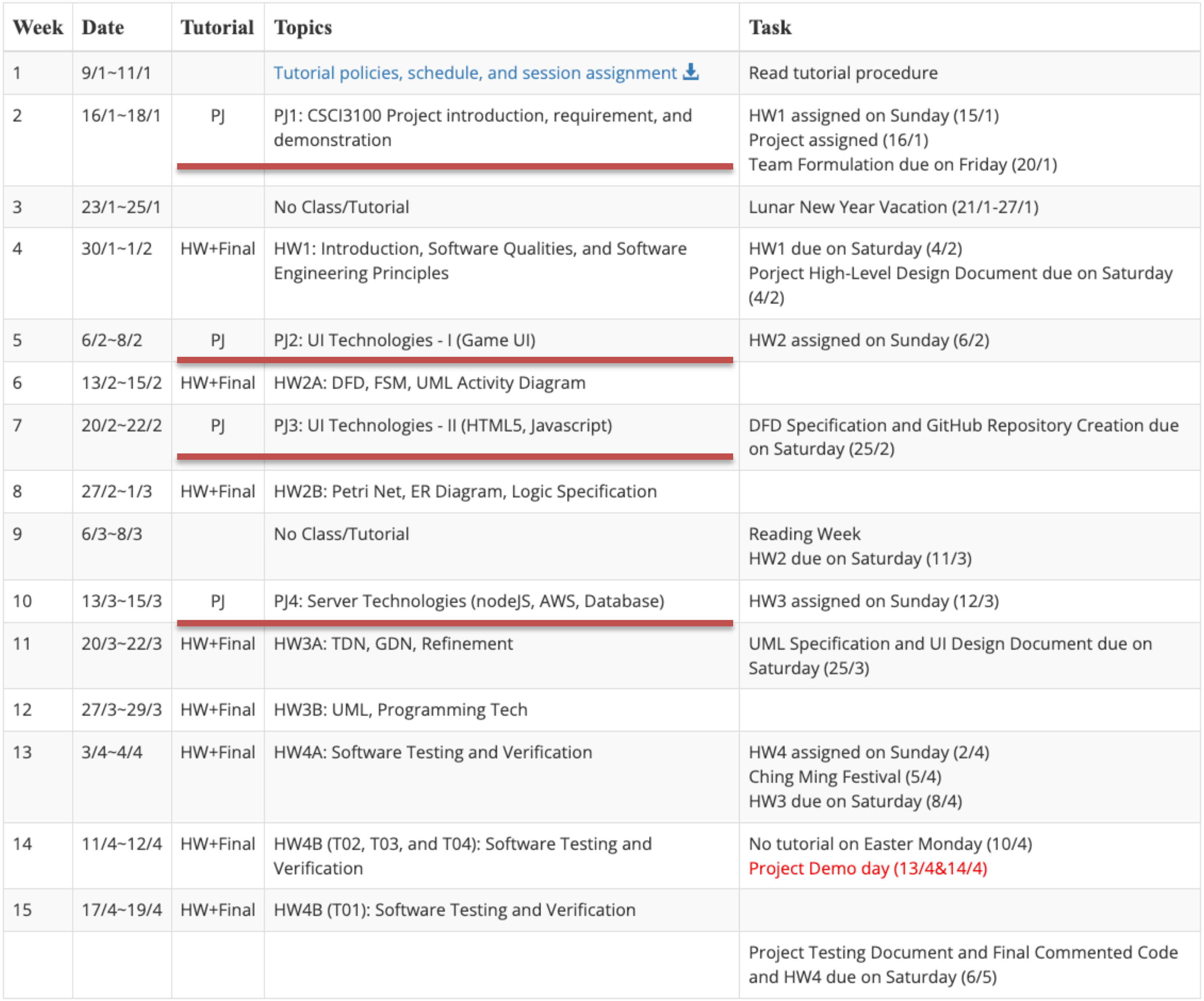
- Please note that designing static HTML web pages is not programming.

- Project tutorials will cover related techniques and tools, such as JavaScript, CCS3, HTML5, Node.js, AWS, Database, etc.

Requirement: Documentation

- One key purpose of this course is that you learn how to do modular design of software and how to document the design using symbolic representations, i.e., UML diagrams.
- The templates are available in the appendix of the project specification.

Tutorial Schedule



Week	Date	Tutorial	Topics	Task
1	9/1~11/1		Tutorial policies, schedule, and session assignment	Read tutorial procedure
2	16/1~18/1	PJ	PJ1: CSCI3100 Project introduction, requirement, and demonstration	HW1 assigned on Sunday (15/1) Project assigned (16/1) Team Formulation due on Friday (20/1)
3	23/1~25/1		No Class/Tutorial	Lunar New Year Vacation (21/1-27/1)
4	30/1~1/2	HW+Final	HW1: Introduction, Software Qualities, and Software Engineering Principles	HW1 due on Saturday (4/2) Porject High-Level Design Document due on Saturday (4/2)
5	6/2~8/2	PJ	PJ2: UI Technologies - I (Game UI)	HW2 assigned on Sunday (6/2)
6	13/2~15/2	HW+Final	HW2A: DFD, FSM, UML Activity Diagram	
7	20/2~22/2	PJ	PJ3: UI Technologies - II (HTML5, Javascript)	DFD Specification and GitHub Repository Creation due on Saturday (25/2)
8	27/2~1/3	HW+Final	HW2B: Petri Net, ER Diagram, Logic Specification	
9	6/3~8/3		No Class/Tutorial	Reading Week HW2 due on Saturday (11/3)
10	13/3~15/3	PJ	PJ4: Server Technologies (nodeJS, AWS, Database)	HW3 assigned on Sunday (12/3)
11	20/3~22/3	HW+Final	HW3A: TDN, GDN, Refinement	UML Specification and UI Design Document due on Saturday (25/3)
12	27/3~29/3	HW+Final	HW3B: UML, Programming Tech	
13	3/4~4/4	HW+Final	HW4A: Software Testing and Verification	HW4 assigned on Sunday (2/4) Ching Ming Festival (5/4) HW3 due on Saturday (8/4)
14	11/4~12/4	HW+Final	HW4B (T02, T03, and T04): Software Testing and Verification	No tutorial on Easter Monday (10/4) Project Demo day (13/4&14/4)
15	17/4~19/4	HW+Final	HW4B (T01): Software Testing and Verification	
				Project Testing Document and Final Commented Code and HW4 due on Saturday (6/5)

Demo:
Selected Previous CSCI3100 Projects