2110336: Team Project
Second Semester, Academic Year 2024

The objective of the team project is to help students become familiar with the major phases of the software development life cycle. Students will gain experience working as part of a software development team. Each team will consist of 8 to 9 members. The entire team is collectively responsible for producing the "deliverables," which include the following items:

Deliverables or Artifacts	**Points**	**Due date**
SRS review report*	3	22-Jan-2025
Product backlog	3	22-Jan-2025
Sprint1 backlog	1	22-Jan-2025
Burndown chart for Sprint1	1	12-Feb-2025
Sprint2 backlog	1	12-Feb-2025
Sprint1 deliverables (software)*	3	13-Feb-2025
Burndown chart for Sprint2	1	12-Mar-2025
Sprint3 backlog	1	12-Mar-2025
Sprint 2 deliverables (software)*	3	13-Mar-2025
Burndown chart for Sprint3	1	26-Mar-2025
Sprint 3 deliverables (software)*	3	27-Mar-2025
Design (Design Document 5% and API is done via Swagger 1%)*	6	9-Apr-2025
Testing Plans and User Manual	8	23-Apr-2025
Demonstration*	10	24-Apr-2025
Git contribution: Issues and contribution	3	All Sprints
Deliver the system on a VM cloud is publicly accessible (during the demonstration)	1	24-Apr-2025
A docker compose file of the term project	1	24-Apr-2025

ส่งงานก่อนเวลา 16:00 น. ของวันที่กำหนดส่ง

*** หักคะแนนนิสิตที่ไม่เข้าร่วมนำเสนอ/ฟังการนำเสนอ คนละ 0.5 คะแนน**

All documents must be typed and submitted to the Course Dropbox on the class's MyCourseville platform in PDF format. Each team must also include the percentage contribution of each team member for every deliverable.

At least some features of the software must be designed and implemented using either Object-Oriented concepts or web development patterns. Each team must demonstrate how the implementation maps to the design.

In each sprint, the deliverables (software) must include a front-end (UI), a back-end (API), and the associated data models (in the database). However,it is not necessary to connect all components during the early sprints.

2110336: SRS Review Report
Due Date: January 22, 2025
SRS Review Checklist
Only review the yellow highlights

ID	Defect Type	Items to Examine	Y/N/NA	Comments
Organization and Structure of the Documentation				
1	Standards	Have appropriate requirements documentation standards been followed?		
2	Standards	Are all figures, tables, and diagrams labeled and referenced?		
3	Standards	Are all terms and units of measure defined?		
4	Standards	Are all requirements written at a consistent and appropriate level of detail?		
5	Standards	Are individual requirements rated (or ranked), with descriptions of priority provided?		
Completeness and Correctness				
6	Correctness	Are all internal cross references to other requirements correct? [For modifiability, minimize cross references.]		
7	Completeness	Are all classes of users described? Are the user characteristics described?		

ID	Defect Type	Items to Examine	Y/N/NA	Comments
8	Completeness	Does the specification include all known customer or system needs? *Are all the tasks the user wants to perform specified?*		
9	Completeness	Is each requirement uniquely and correctly identified?		
10	Completeness	Does each functional requirement specify input and output, as well as function, as appropriate?		
11	Completeness	Have all dependencies on other systems been identified? (applications or application interfaces, databases, communications subsystems, networking, etc.)		
12	Completeness	Are user documentation and training requirements addressed?		
13	Completeness	Are the hardware and software environments specified?		
14	Completeness	Have all derived requirements been included? (those implied by the system or software requirements, generally constraints on development or verification)		

ID	Defect Type	Items to Examine	Y/N/NA	Comments
15	Completeness	Has full life cycle support been addressed, including maintenance?		
16	Completeness	Are any design or implementation constraints described?		
17	Completeness	Have non-functional requirements or all quality attributes (characteristics) been properly specified (i.e. efficiency, flexibility, interoperability, maintainability, portability, reusability, usability, availability)		
18	Completeness	Have the human interface requirements been addressed? Are they correct?		
19	Completeness	Are all external hardware, software, and communication interfaces defined? Are they correct?		
Consistency, Clarity, and Verifiable				
20	Consistency	Does the specification agree with all relevant higher-level documents?		
21	Consistency	Are the requirements free of duplication and conflict with other requirements?		

ID	Defect Type	Items to Examine	Y/N/NA	Comments
22	Consistency	Is each requirement written in consistent, clear, concise language?		
23	Clarity	Does each requirement have only one interpretation? If a term could have multiple meanings, is it defined?		
24	Verifiable	Is each requirement verifiable by testing, demonstration, review, or analysis?		
25	Verifiable	Are there measurable acceptance criteria for each functional and non-functional requirement?		
Traceability				
26	Consistency	Is each requirement traceable to use case?		
27	Consistency	Is each use case traceable to a specified sequence diagram?		
28	Consistency	Are all sequence diagrams and their messages (methods and signatures) traceable to the class diagram (methods)?		
Special Issues				
29	Other	Is each requirement in scope for the project?		
30	Other	Are all requirements actually requirements, not design or implementation solutions?		
ID	**Defect Type**	**Items to Examine**	**Y/N/NA**	**Comments**
31	Other	Are the time-critical functions identified, and timing criteria specified for them?		

32	Other	Are all significant consumers of scarce resources (memory, network bandwidth, processor capacity, etc.) identified, and is their anticipated resource consumption specified?		
33	Other	Have internationalization issues been adequately addressed?		

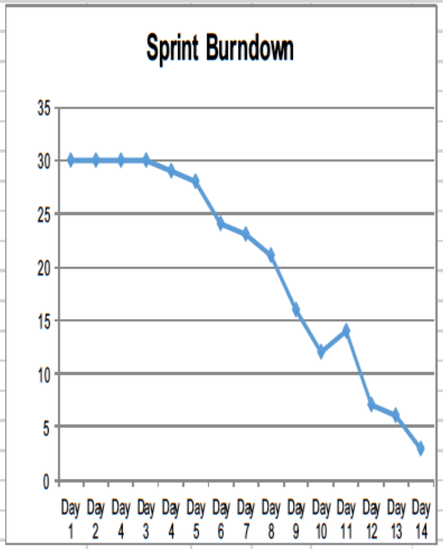
2110336: SCRUM Deliverables
Product Backlog
Due Date: January 22, 2025
Presentation Date: January 23, 2025

Product backlog

EPIC ID	**EPIC Name**	**User Story ID**	**User Story**	**Estimate (User Story Points)**	**Estimate (Man-Hour)**
EPIC1	EpicName1	US1-1	As a <role>		
			I can <activity>		
			so that <business value>		
		US1-2			
EPIC2	EpicName2	US2-1			

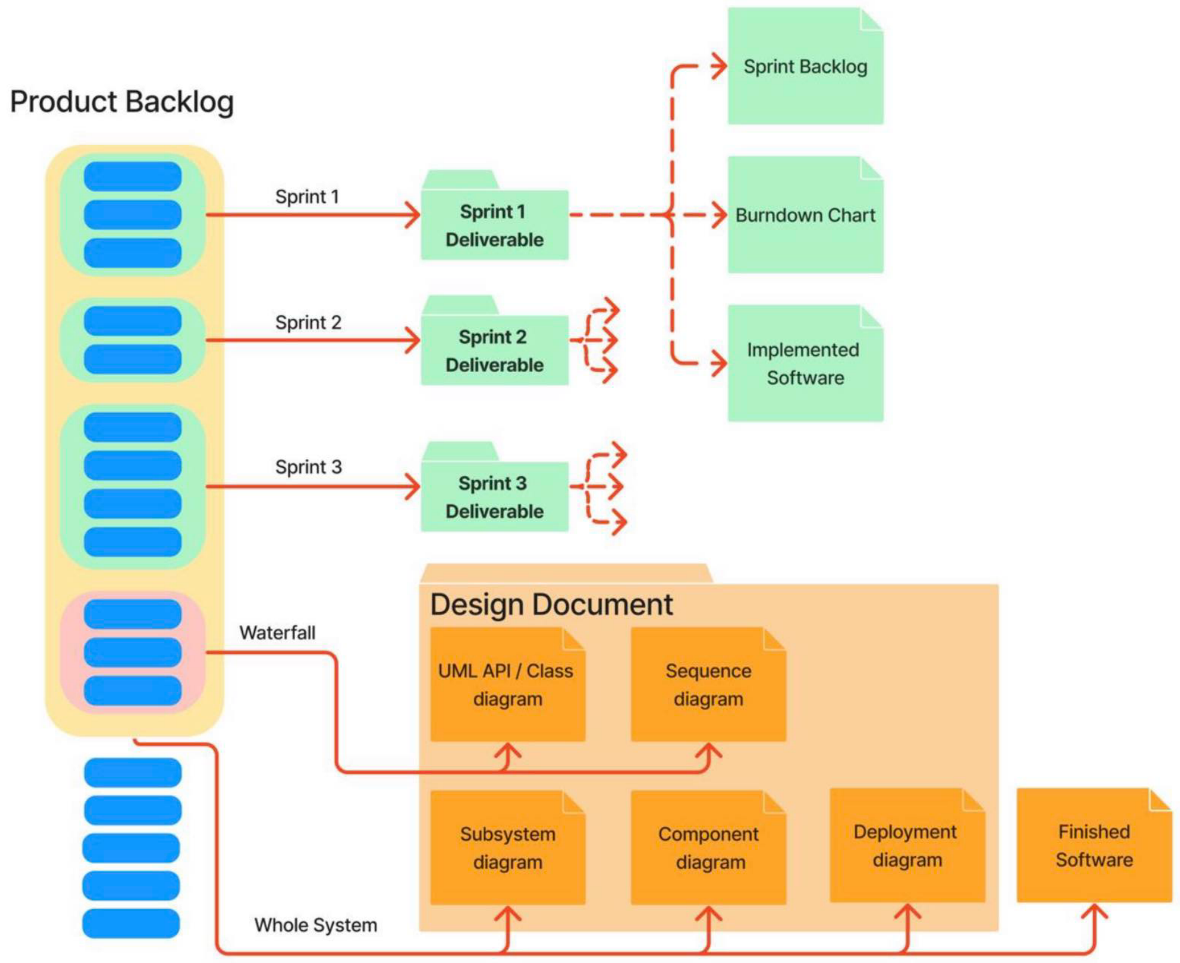
2110336: SCRUM Deliverables
Sprint backlog and Burndown chart

Sprint ID	EPIC ID	EPIC Name	User Story ID	User Story	Task	Volunteer	Status	Original Estimate	Day 1	Day 2	Day 4	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12	Day 13	Day 14	Sprint Review
Sprint1	EPIC1	EPIC Name1	US1-1					4	4	4	4	4	3	3	0	0	0	0	0	0	0	0	0	0
								4	4	4	4	4	4	4	3	2	1	0	0	0	0	0	0	0
								6	6	6	6	6	6	6	6	6	6	4	1	0	0	0	0	0
								4	4	4	4	4	4	3	3	3	2	0	0	0	0	0	0	0
								3	3	3	3	3	3	3	3	3	3	3	2	0	0	0	0	0
								3	3	3	3	3	3	3	3	3	3	3	3	2	1	0	0	0
								6	6	6	6	6	6	6	6	6	6	6	6	12	6	6	3	0
Sprint1	EPIC2	EPIC Name2	US2-1																					
							Todo	ยังไม่select มาทำ																
							Develop	กำลัง develop (design and code)																
							Test	กำลัง test																
							Done	ทำเสร็จ																
					Total			30	30	30	30	30	29	28	24	23	21	16	12	14	7	6	3	0



2110336: Waterfall Deliverables
The Design Document
Document Due Date: April 9, 2025
Presentation Date: April 10, 2025

1. Cover Page
    - title of the project
    - a list of the authors names (indicate the leader)
    - contribution of each team member to the document
    - the date
2. Architecture



- For the whole system (S1+S2+S3+Waterfall)
    - Subsystem diagram
    - Component diagram
    - Deployment diagram
- For the selected features (Waterfall)
    - Sequence diagram (one use case)
    - Each group must submit **either**
        - Class diagram including Attributes, Operations, Visibility (if using OO design) **or**
        - UML API diagram (if using web development pattern)

3. Access Control Table **or** API CRUD

2110336: Final Deliverables
The Test Plan and User Manual
Document Due Date: April 23, 2025
Presentation Date: April 24, 2025

1. Cover Page
   - title of the project
   - a list of the authors names (indicate the leader)
   - contribution of each team member to the document
   - the date
2. Black-box Test Plan

- Choose 4 main system functions to be tested
- For each system function
    - o write input equivalence classes
    - o generate test cases to cover all input equivalence classes
  - A test case includes items as follows:
    - o Test Case ID
    - o Inputs
    - o Expected Outputs
    - o Actual Outputs
    - o Equivalence Classes Covered
- Write Test Coverage Matrix

Function ID	Function Name	Test Cases ID	Status
1		87,88,89	P, F, P
2		81-88,102	
3			
4		103-106	

3. User manual of your system
4. Report any limitations in your implementation, including functions that could not be coded, functions with significant errors, or any deviations from the design document.

Project Demonstration

1. Demonstrate the system's operations for every function, and show the code that corresponds to the design in the Class Diagram (Object-Oriented design) or the UML API Diagram.
2. Perform manual black-box testing on two main functions by inputting 2–3 test cases from the documentation.
3. Conduct black-box testing for two additional main functions using the Robot Framework (or another test automation framework).
4. Test one function using test cases with a unit testing tool (such as JEST), ensuring the function meets the 100% statement coverage criterion.