# A7 - System documentation

for

# Crowd Quizmaker

**Version 1.1**

**Prepared by 0N3 N16H7 PR0J3C7**

Suwat Inkaew 610610521
Kritsanaphong Tepweerakul 630610714
Kitpisan Tanngan 630610716
Chayanon Pitak 630610724
Nadtaphong Jandaboot 630610743
Woranut Kitchakan 630610760

https://github.com/ChayanonPitak/261361-Project/

# Revision History

| Name | Date | Reason for changes | Version |
|------|------|--------------------|---------|
| Chayanon | 23 Mar 2023 | Complete of Sprint#2 | 1.0 |
| Chayanon | 30 Mar 2023 | Add A4 & A6 | 1.1 |

# Table of Contents

# 1 A1 - Project proposal

## 1.1 Team profile

- **Chayanon Pitak** as Project Manager, System Analyst and Developer
  **Skills**

  - **Web application development** on JavaScipt, Typescript, React.js, Solid.js, PHP, TailwindCSS.
  - **Software development** on Java, C, C++.
  - **Game development** on Unity using C#.

  **Experience**

  - Mostly college projects and personal projects.

- **Kritsanaphong Tepweerakul** as Developer and Business Analyst
  **Skills**

  - **Web application development** on JavaScipt, Typescript, React, TailwindCSS, PHP, Laravel.
  - **Software development** on C, C++, C#, Java, Python.
  - **Game development** on Spring boot/React Web-based Game, Unity.

  **Experience**

  - SFML 2D-RPG Game.
  - CARIN. (Strategy-Game Project)
  - Gogoboard Automatic Watering.
  - Store Website.

- **Woranut Kitchakan** as Designer
  **Skills**

  - **Web application development** on JavaScipt, Typescript, React, TailwindCSS, Laravel.
  - **Software development** on C++, Java.
  - **Game development** on C++, Spring boot/React Web-based Game.

  **Experience**

  - SFML Platform games.
  - CARIN. (Strategy-Game Project)
  - Gogoboard Automatic clothes drying machine.

- **Kitpisan Tanngan** as Developer and Designer **Skills**

  - **Web application development** on JavaScipt, Typescript, React, TailwindCSS, Laravel.
  - **Software development** on C++, Java.
  - **Game development** on C++, Java.

  **Experience**

  - Turn-base Game Project.

- Gogoboard Automatic roof for plants.

- **Nadtaphong Jandaboot** as Tester and Designer
  **Skills**

  - **Web application development** on TailwindCSS.
  - **Software development** on C++.
  - **Art and Editing** on Photoshop.

  **Experience**

  - Mostly a art designer and sometimes front-end developer in projects.

- **Suwat Inkaew** as Tester **Skills**

  - **Web application development** on a little bit of JavaScipt, Typescript, React, TailwindCSS.
  - **Software development** on C, C++ and a little bit of C#, Java.

  **Experience**

  - SCRATCH Game.
  - CARIN(Tower Defense Game).
  - Review Website.
  - Card Game Project.

- **Nadtaphong Jandaboot** as Tester and Designer

## 1.2   Crowd quizmaker

Students and lecturers generate quiz and answers. Quizzes can be edited, commented, voted and rated (e.g. quiz-maker)

## 1.3   Problem statement

Nowadays "Social network" is very popular for people to hang out on, they can see what their follower are doing, they can interact with follower and they can share their own content.

But there is no "Examination Social Network" yet. Which can be a way to evaluate student's knowledge and let them to compete with each other that also fun for the student. For example, in the past Internet and Online Community course (261111) have a traditional examination, which is boring and not interactive. But if we have a "Examination Social Network" that can be a way to evaluate student's knowledge and let them to compete with each other that also fun for the student.

## 1.4 Solution

### 1.4.1 Facebook group
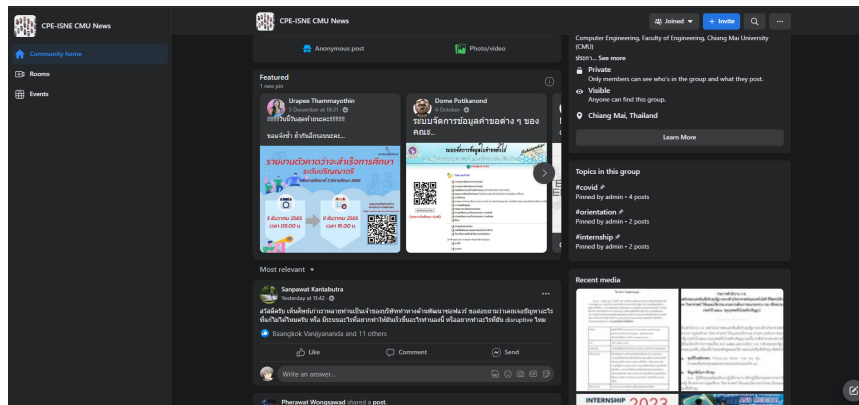


**Facebook group** is a group platform created by Meta. It let people with same interest to gather or lecturer can communicate with student. Also it is a way to let people create post (including quiz) and interact with each other. But it cannot evaluate student performance nor quiz quality and managing posts into topics/quizes is very hard or impossible to do.

### 1.4.2 Quiz-maker
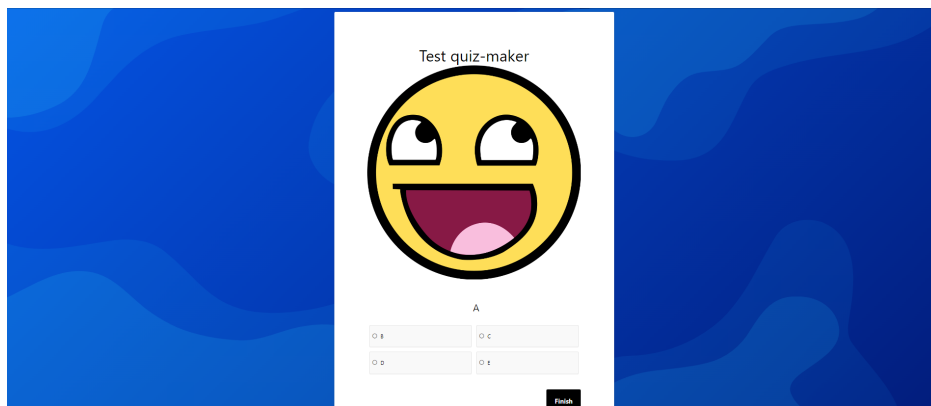


**Quiz-maker** is a quiz creation platform. User can create quizzes of their own topic and share to others. It have great evaluation but lack of interaction and quizzes quality review from other users is impossible.

### 1.4.3 Kahoot!
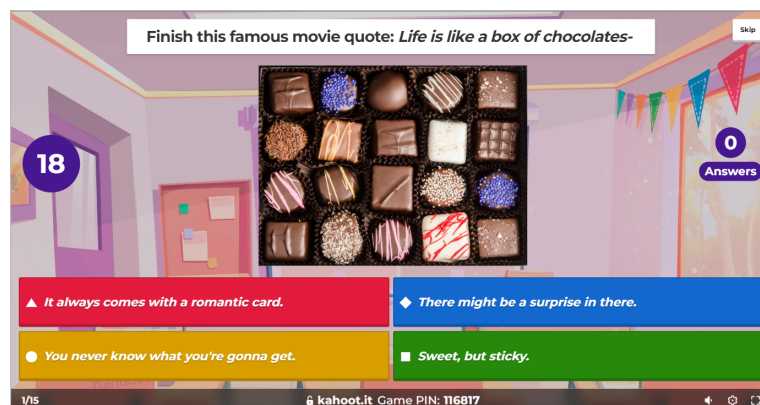


**Kahoot** is a game-based learning platform. It let lecturer create questions according to topics and student can syncronously answer the questions with intensive environment. It have great evaluation and fun. But students cannot contribute their own quiz to existing quizzes.

### 1.4.4 Learning managament syatems (LMS) - Moodle, Blackboard, Canvas, etc.
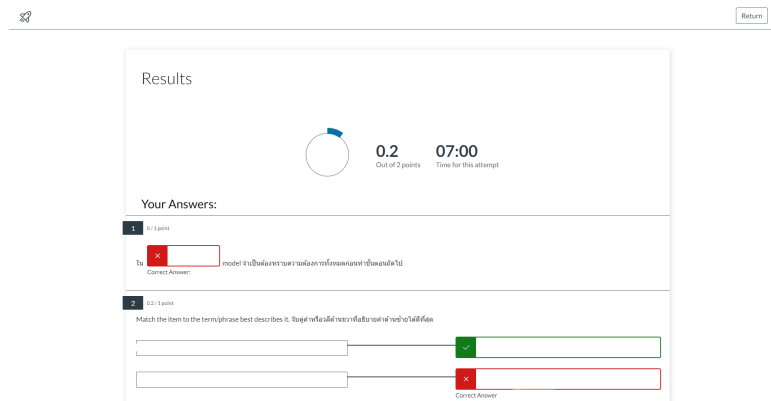


**Learning management system (LMS)** is an application that automates the administration, tracking, and reporting of training events.[1] It have great evaluation and grading, but quizzes looks like traditional examination.

---

[1] Ryann K. Ellis, Field Guide to Learning Management Systems, Learning Circuits, 2009, p.1

## 1.5 Contribution

In general, quiz is created by one person or one group. It makes the number of quiz and variety of quiz is low. We have an inspiration of Facebook group which everyone in the groups can contribute to a topic, which will increase the variety and number of quiz. The process of creating quiz is as follows: define questions, define choices, define correct answers, have a system that can comment or vote on the quiz created by others to review the quality and correctness of the quiz, have a system that can evaluate the quality of the quiz.

### 1.5.1 Comparison to existing solutions

- Facebook groups

|  | Facebook group | Our solution |
|---|---|---|
| Quiz creation | Creating posts | Creating quizzes with defined format |
| Quiz contribution | Everyone in group or group moderator approval | Defined by subject creator |
| Quiz validation | - | Pre-defined format |
| Quiz quality review | Reactions and Comments | Pre-defined rubrics |
| User evaluation | Manually | User dashboard |

- Quiz-maker

|  | Quiz-maker | Our solution |
|---|---|---|
| Quiz creation | Creating quizzes with defined format | Creating quizzes with defined format |
| Quiz contribution | Quiz creator | Defined by subject creator |
| Quiz validation | Pre-defined format | Pre-defined format |
| Quiz quality review | - | Pre-defined rubrics |
| User evaluation | User dashboard | User dashboard |

- Kahoot!

|  | Kahoot! | Our solution |
|---|---|---|
| Quiz creation | Creating quizzes with defined format | Creating quizzes with defined format |
| Quiz contribution | Quiz creator | Defined by subject creator |
| Quiz validation | Pre-defined format | Pre-defined format |
| Quiz quality review | Stars rating | Pre-defined rubrics |
| User evaluation | User dashboard | User dashboard |
| Syncronous activity | ✓ | ✓ |

- Learning management system

|  | LMS | Our solution |
|---|---|---|
| Quiz creation | Creating quizzes with defined format | Creating quizzes with defined format |
| Quiz contribution | Quiz creator | Defined by subject creator |
| Quiz validation | Pre-defined format | Pre-defined format |
| Quiz quality review | - | Pre-defined rubrics |
| User evaluation | User dashboard | User dashboard |

## 1.6 Stakeholder and User groups

- **Product Owner**

  - Professor Kampol Woradit as Internet and Online Community course lecturer.
  - Professor Sakgasit Ramingwong as Internet and Online Community course lecturer.

- **Study purpose**

  - Lecturers
  - Students

- **Entertain purpose**

  - Quiz Makers
  - Who want to take advantage from their free time
  - Who is interested in specific purpose
  - Content creater who want to created content based on fan-quizzes

## 1.7 Technology feasibility study

We will create web application which is cross-platform friendly that can be use on mostly any device.

### 1.7.1 Tool and resources used

- Figma for UI/UX design.

- Swagger for API design.

- diagrams.net for database design.

- HTML/CSS for web markdown and styling.

- Typescript for backend

- React.js for frontend framework.

- MySQL for database system.

- Prisma for database orm.

## 1.8 Conclusion

We will create cross-playform web application that's looks like combination of Facebook groups and LMS, which lets user gather in groups and create quizzes together. Also users can review and rated quizzes quality.

We will separete into two user groups.

- For **Lecturer and student**, lecturer can create and subject topic and let student create quizzes related to that topic. After that student will take each other quizzes to test their knowledge and rate their quizzes quality. This method make the quizzes become larger and more accurate for that subjects that student also have fun.

- For **Fans**, they can create quizezs related to their favorite topic and let others to contribute that topic.

# 2 A2 - Software Requirement Specification

## 2.1 Introduction

### 2.1.1 Purpose

Lecturer team of 261111 - Internet and Online Community, Chiang Mai University have created this course to create realize in technology, internet and online community and proper behaviour of using it. Originally, the course itself are using typical way of evaluate the understanding - assignments and exams which is one-way activities. The lecturer team have decided to create a new way of evaluate the understanding of the students by creating a quiz application that students also contribute the quiz creation from the knowledge they have learned from the course and try to answer and review the contributed quizzes. The application will be called Crowd Quizmaker.

### 2.1.2 Scope

Our software (Crowd Quizmaker) will be a cross-platform (Windows, MacOS, iOS, Android) web application that lecturer (or quiz admin) can create a quiz topic and students (or anyone - as quiz contributor) can contribute to a quiz - create, answer and review. Any users can be quizzes admin and/or quiz contributor and quizzes topic doesn't need to be academic purpose. Quizzes admin can specify what can quizzes contributor do.

### 2.1.3 Product Overview

**Product Perspective**

This product is a web application that can be accessed from any devices that have internet connection. The application will be a cross-platform (Windows, MacOS, iOS, Android) web application. The application is a quiz platform that any one can contribute to quizzes. The quizzes can be academic or non-academic.

**Product Functions**

The application consits of two part - Central system which provide the API, quizzes database storage and quizzes manager. And user client that allow any users to create quiz topic, manage quiz topic - moderate, manage, view and evaluate quiz content, contribute to quizzes - create, answer and review, and add support to Canvas LMS.

**User Characteristics**

The user classes of this systems will be:

1. Administrator: Lecturer team of 261111 - Internet and Online Community, Chiang Mai University or any one who have the permission to manage the system. Responsible for manage, support and operate whole systems - both central system and clients.

2. Course Lecturer: Any course lecturer who wish to use this system. Responsible for manage the quizzes topic, moderate quizzes content, evaluate quiz topic and/or Canvas LMS intregration.

3. Course student: Student from courses. Responsible for Canvas LMS connection and contribute to quizzes from courses - create answer and review.

4. Quiz Admin: Anyone who want to create quizzes topic. Responsible for create/manage the quizzes topic, moderate quizzes content and evaluate quizz topic.

5. Quiz Contributor: Anyone who want to contribute to quiz topic. Responsible for create, answer and review quizzes from quiz topics.

**Limitations**

Currently we do not have any limitations from the requirements of stakeholders, but it may including but not limited to hardware performance limitstions, storage limitations, internet connection limitations, specific course content, local regulations and etc. We will update this subsection if we have any exact limitations to update.

### 2.1.4 Definitions

- **Central system or server**: The API provider, quizzes database storage and quizzes manager. Only accessible by administrator.

- **Client**: The user interface that allow any users to interact with our systems - Manage and moderate quiz topics and contribute to quizzes.

- **Quiz**: A set of questions and answers that can be answered by users.

- **Quiz Topic**: A topic that contain quizzes.

- **Quiz Admin**: An user that can create/manage quiz topic, moderate quizzes content and evaluate quiz topic.

- **Quiz Contributor**: An user that can contribute to quiz topic - create, answer and review quizzes.

- **Course lecturer or lecturer**: A course lecturer which have same permissions as Quiz Admin.

- **Course student or student**: A student which have same permissions as Quiz Contributor.

## 2.2  References

ISO/IEC/IEEE 29148:2018
https://ieeexplore.ieee.org/document/8559686
Systems and software engineering – System and software requirements engineering – System requirements specification


CPE111 - Internet and Online Community facebook pages
https://www.facebook.com/CPE111/


Canvas Learning Management System
https://www.instructure.com/en-au/canvas/

## 2.3 Specific Requirements

### 2.3.1 External Interface

**System Interfaces**

The application will retrieve the registration information for the required course from "canvas.cmu.ac.th" in the form of JSON. This will provide the registration information for the application.

**User Interfaces**

The user interface will be simple to understand for all the user that uses this application. We will design a new user interface for our application using our experience and knowledge that doesn't consist of any standard. The reason is to eliminate the difficulty of the user's understanding of our application interface.

**Hardware Interfaces**

No extra hardware interfaces are needed.

**Software Interfaces**

The application will use the GraphQL API from Canvas.

**Communications Interfaces**

The application will use the HTTP protocol to communicate with the client. So, an internet connection is required.

### 2.3.2 Functions

The system shall

1. **Let user create quiz topic** which contains topic name, tags(optional), contributors(optional), format(multiple choices and/or text area), and distribution(question only, answer only or both).

2. **Let quiz topic admin manage quiz topic** on topic name, contributors, format, and distribution and delete whole quiz topic.

3. **Let quiz topic admin moderate quizzes content** on contributors' question, answer and review.

4. **Let contrubutor create question to the quiz topic** based on quiz topic distribution and format. Contributor shall define the question, tag(depends on admin), answer(s) and correct answer(s).

5. **Let contrubutor attempt to answer and review the question to the quiz topic** based on quiz topic distribution. System shall random the question based on distribution settings. User shall answer the question and review the question - rate, comment and/or report. (only quiz topic admin can see the review)

6. **Evaluate the user's quiz attempt** which visualize the result of the quiz attempt. The result contains correct answer count on that attempt.

7. **Let quiz topic admin evaluate the quiz topic** which visualize the result of the quiz attempt. The result contains correct answer count on questions and questions' review.

### 2.3.3 Usability Requirements

The system shall be able to operate normally without any errors or server issues. Also, it shall be simple to use for every user.

### 2.3.4 Performance Requirements

- **Respond Time**: The system shall have a response time of fewer than 10 seconds for topic-related requests, except for adding contributors shall not have a responce time greater than 30 seconds.

- **Throughput**: The system shall be able to handle 500 simultaneous users and shall be able to handle at least 100 requests per second.

- **Memory usage**: The system shall use no more than 1 GB of memory under normal conditions.

- **CPU usage**: The system shall require atleast 4 core of CPU under normal conditions.

- **Storage**: The system shall be able to store up to atleast 1 TB of data.

- **Avalibility**: The system shall always be available to the user all the time.

### 2.3.5 Logical Database Requirements

1. The database shall be able to store user information and quiz data.

2. The database shall always be able to provide data for the server most of the time.

3. The database shall be able to ensure that each role of user can access their interface by their role.

### 2.3.6 Design Constraints

- **Time**: This application must be complete within 2-3 weeks.

- **Resource**: This application project team has 6 members.

- **Scalability**: The system must be design to support a minimum of 500 simultaneous users.

- **Deployment**: The system must be deployable to server-based and client-based infrastructure.

### 2.3.7 Software System Attributes

- **Reliability**: The system shall be able to perform an intended function without unexpected errors.

- **Maintainability**: The system shall be easy to maintain and update.

- **Portability**: The system shall be able to run on multiple operating system, including Window, Mac, Android and iOS.

### 2.3.8 Supporting Information

1. The application shall be able to run on every modern browsers.

2. The application shall not send any personal data of the user to the internet or an unknown source.

## 2.4 Verification

### 2.4.1 External Interface

**System Interfaces**

- **Fetch from canvas**: After retrieving the course information, we will check the number of people on the course that are the same number people of information we have retrieved.

- **Import manually**: After imported manually, we will check that people are imported are really in this course.

**User Interfaces**

We will be using "User Acceptance Testing" in user interface by taking users to use the interface to check the simplest of the website.

**Software Interfaces**

We will test by checking whether the information is correct or not using "System Testing".

**Communications Interfaces**

We won't test because HTTP protocal are standard system and already Verificate.

### 2.4.2 Functions

We will be using "User Acceptance Testing" by finding the tester to test the function that is working properly or not.

### 2.4.3 Usability Requirements

Tester will test the app and send feedback about user experience.

### 2.4.4 Performance Requirements

- **Respond Time**: Tester will test by reckon system's response time.

- **Throughput**: Tester will test by sending requests to system.

- **Memory usage**: Tester will test by performance monitoring while running application.

- **CPU usage**: Tester will test by performance monitoring while running application.

- **Storage**: Tester will test by performance monitoring while running application.

- **Avalibility**: Tester will test by uptime monitoring.

### 2.4.5 Logical Database Requirements

1. Tester will test by send requests about user information and quiz data.

2. Tester will test by uptime monitoring about data.

3. Tester will test by access each rolls and test.

### 2.4.6 Design Constraints

- **Time**: Verification by scrum meeting, github Kanbanboard to check progress on implementation.

- **Resource**: Have a verification on skills and progress done by each members.

- **Scalability**: Verification by sending requests to system.

- **Deployment**: Verification by show connection between server and client.

### 2.4.7    Software System Attributes

- **Reliability**: Verification by test at most as possibe expected errors.

- **Maintainability**: Verification by change some part and it will done by few steps and don't have any causes to another parts.

- **Portability**: Verification by testing application on multiple operating system.

### 2.4.8    Supporting Information

1. Verification by testing application on multiple modern browsers.

2. Verification by penetration test.

- **Reliability**: Verification by test at most as possibe expected errors.

- **Maintainability**: Verification by change some part and it will done by few steps and don't have any causes to another parts.

# 3  A3 - Design Specification

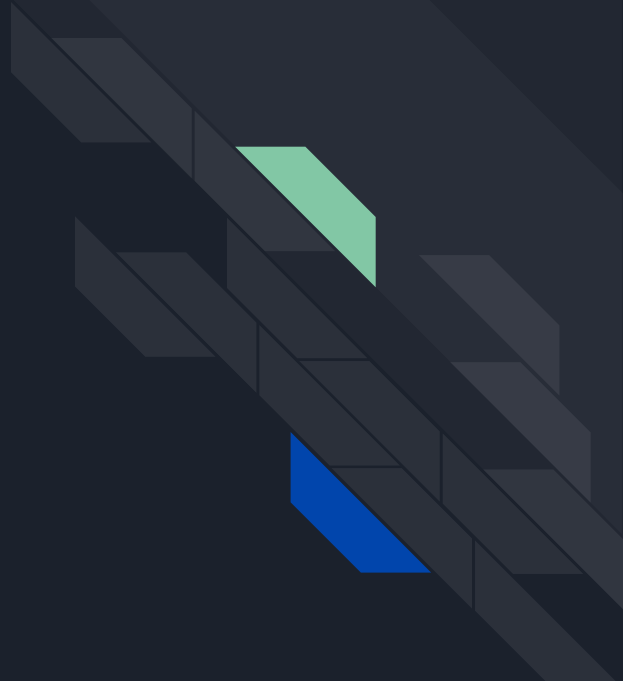Due to size issue, please visit here for the design specification.
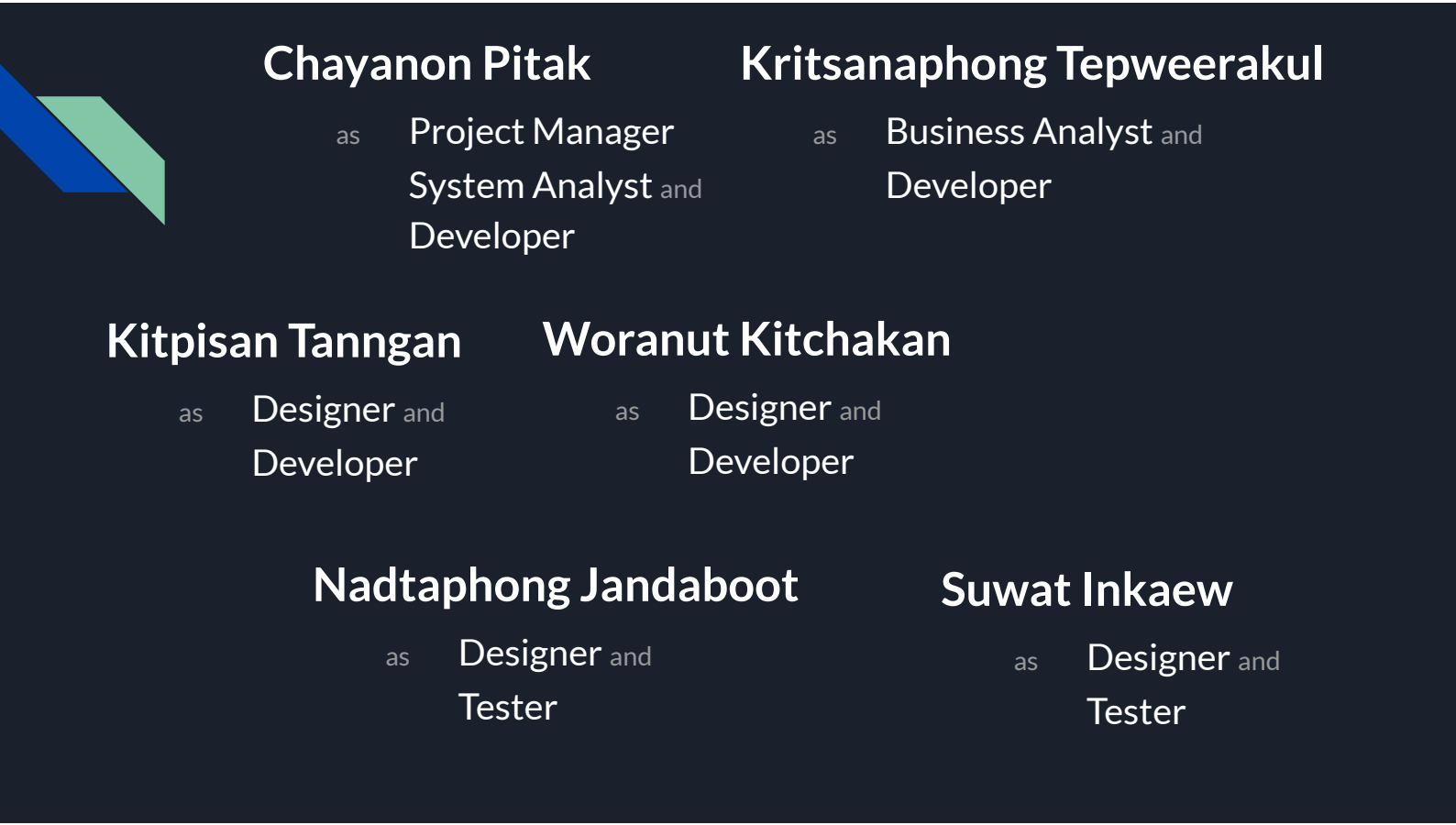
# 4 A4 - Sprint #1

# Crowd Quizmaker (CQ)

Presented by 0N3 N16H7 PR0J3C7

# Members

# Chayanon Pitak

as Project Manager System Analyst and Developer

# Kritsanaphong Tepweerakul

as Business Analyst and Developer

# Kitpisan Tanngan

as Designer and Developer
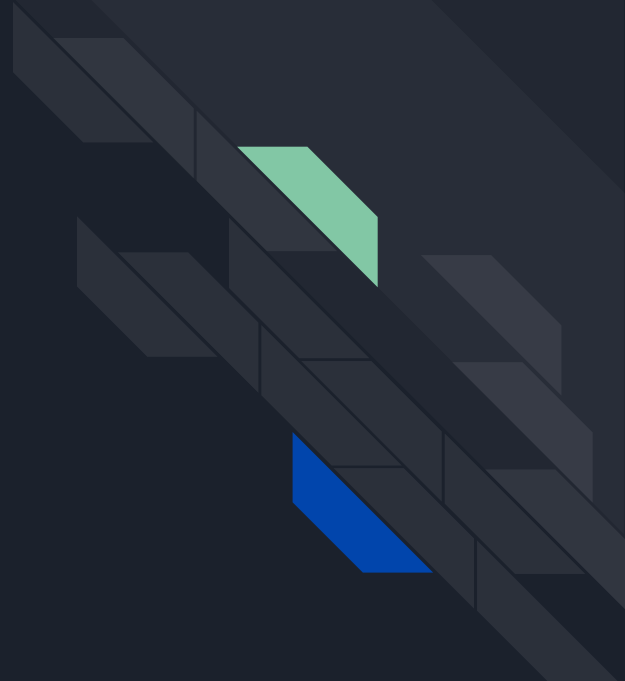
# Woranut Kitchakan

as Designer and Developer

# Nadtaphong Jandaboot

as Designer and Tester

# Suwat Inkaew

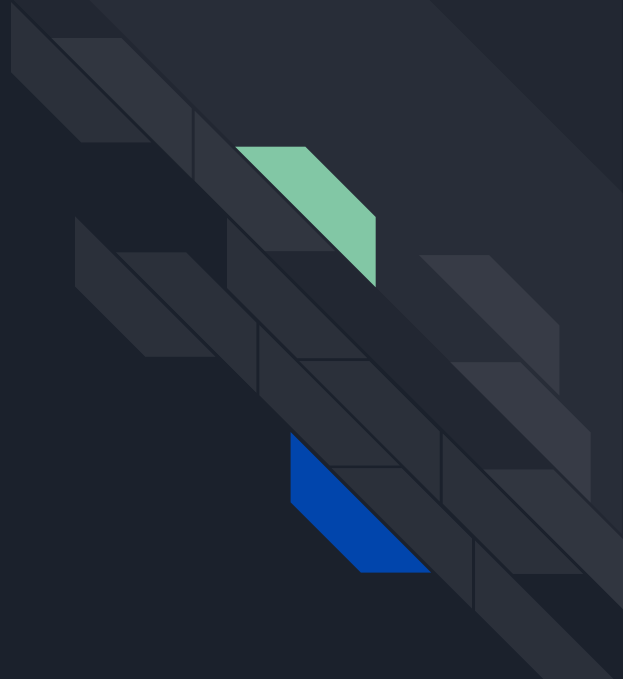as Designer and Tester

# Our Project

# Scope
(Sprint #1)

Stakeholders are able to know the basic concept of the Desktop application including **Topic creation**, **Question creation**, and **Quiz attempt**.

# Features
## (Sprint #1)

- Basic authentication
- Create topic
- Create questions
- Attempt the quizzes
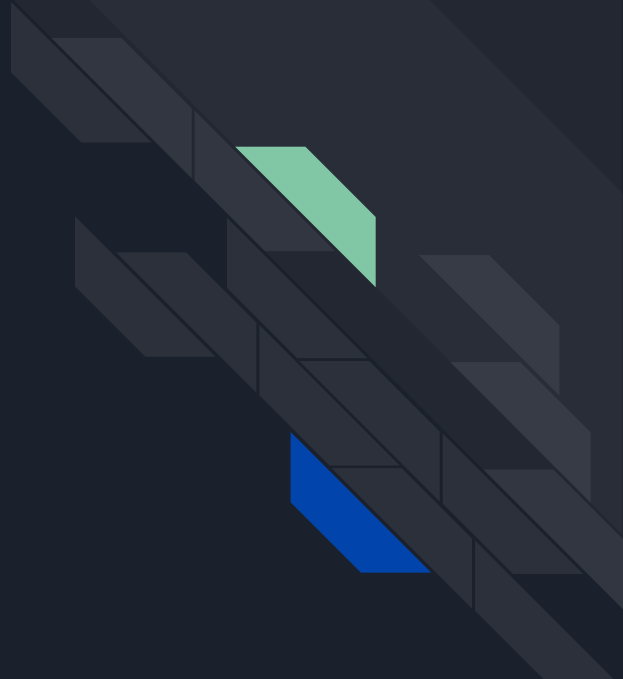
# Scope
## (Sprint #2)

Stakeholders are able to know the complete concept of
The cross-platform application including more details on
Sprint#1 and **Complete Authentication, Usage on large scale,
Quiz topic management and moderation, Attempt review and evaluation.**
After this sprint it should be able to handle basic usage.
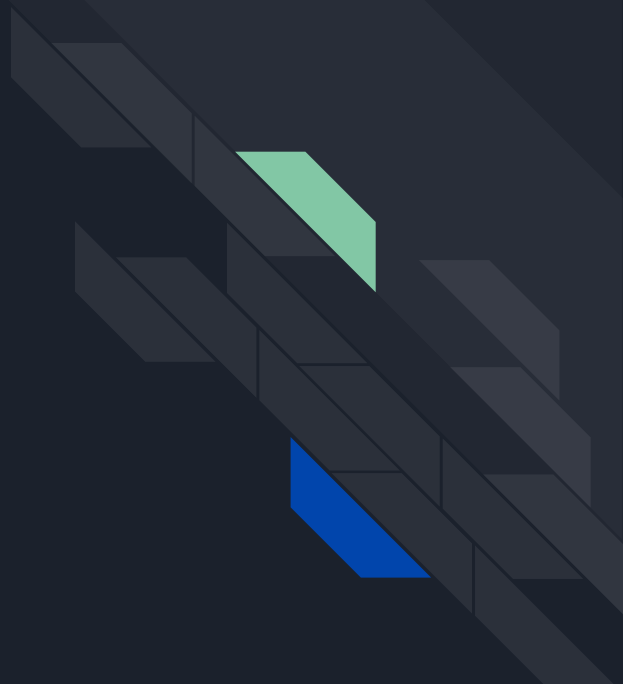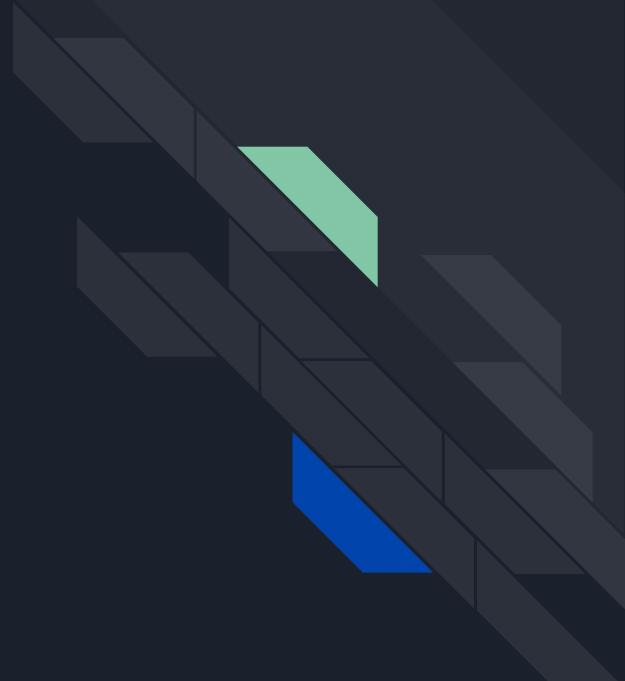
# Features
## (Sprint #2)

- Complete authentication (Basic + CMU OAuth)
- Create, manage, and moderate topic
- Create and manage questions
- Attempt, review and evaluate the quizzes
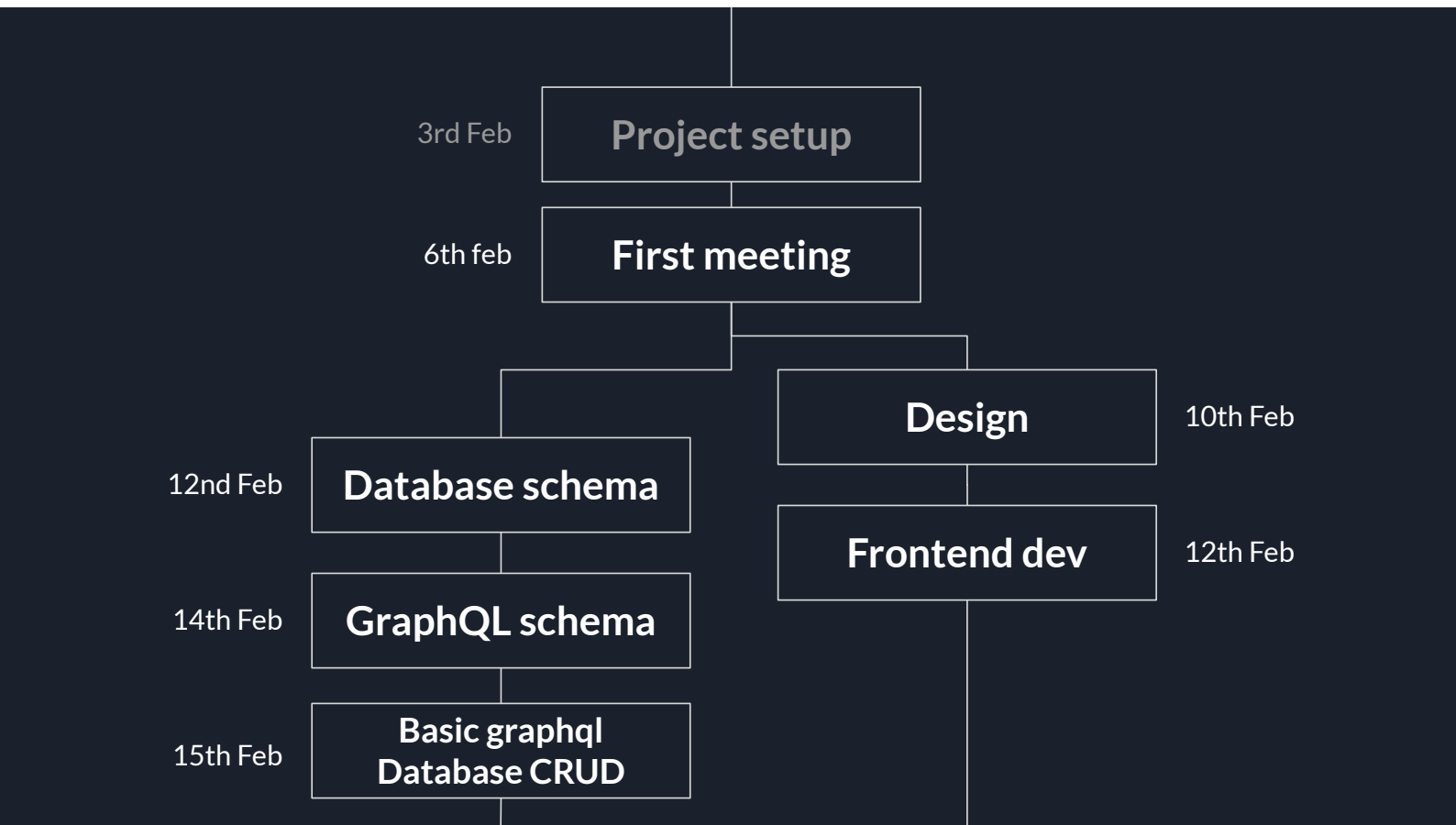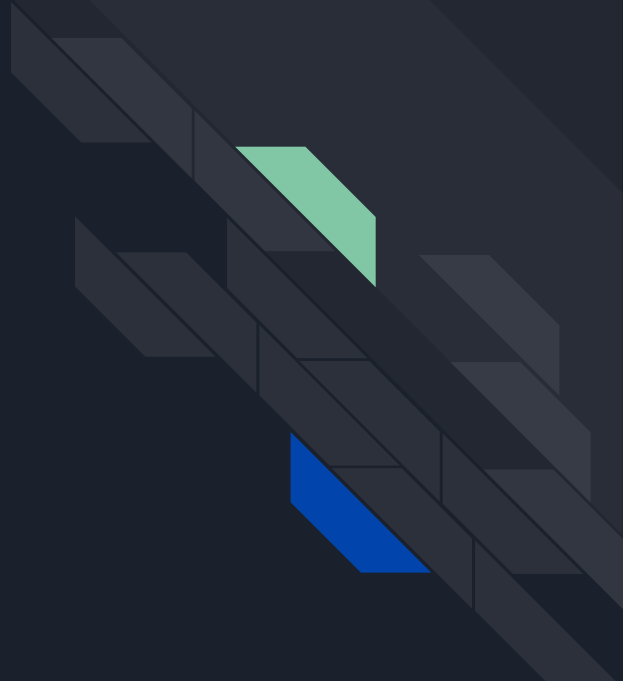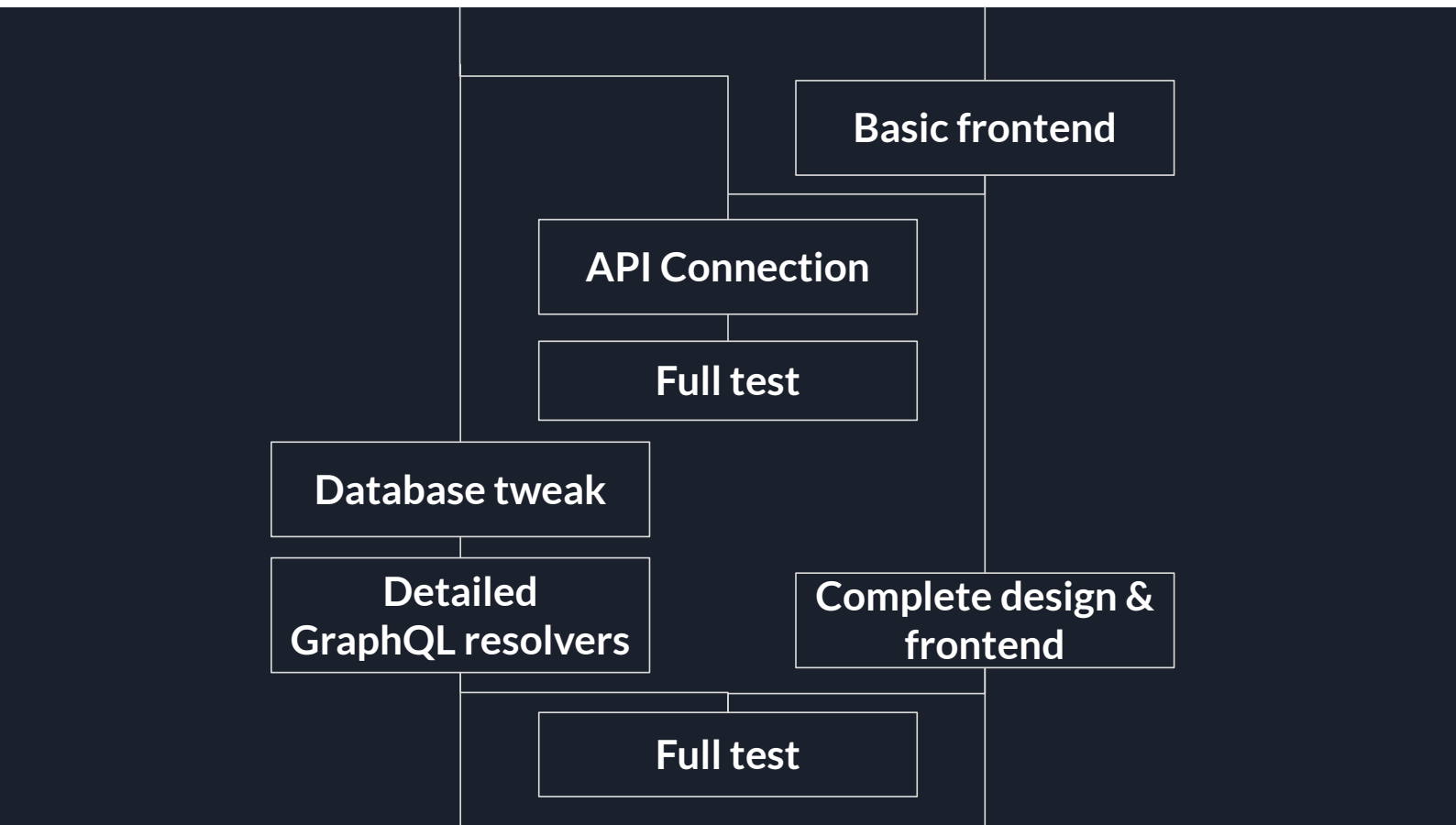- Import/Export data between LMS

Demo

# Retrospection

```
                        |
3rd Feb      ┌──────────┴──────────┐
             │   Project setup     │
             └──────────┬──────────┘
6th feb      ┌──────────┴──────────┐
             │   First meeting     │
             └──────────┬──────────┘
             ┌──────────┴──────────┬──────────┐
             │                               │
             │                     ┌─────────┴─────────┐
             │                     │     Design        │   10th Feb
             │                     └─────────┬─────────┘
12nd Feb  ┌──┴───────────────┐     ┌─────────┴─────────┐
          │ Database schema  │     │   Frontend dev    │   12th Feb
          └──┬───────────────┘     └─────────┬─────────┘
14th Feb  ┌──┴───────────────┐               │
          │ GraphQL schema   │               │
          └──┬───────────────┘               │
15th Feb  ┌──┴───────────────┐               │
          │  Basic graphql   │               │
          │  Database CRUD   │               │
          └──────────────────┘               │
```

- 3rd Feb — **Project setup**
- 6th feb — **First meeting**
- 10th Feb — **Design**
- 12th Feb — **Frontend dev**
- 12nd Feb — **Database schema**
- 14th Feb — **GraphQL schema**
- 15th Feb — **Basic graphql Database CRUD**

# Plan

Basic frontend

API Connection

Full test

Database tweak

Detailed
GraphQL resolvers

Complete design &
frontend

Full test

# 5  A5 - Test Document

Software Name : Crowd Quizmaker Team name: 0N3 N16H7 PR0J3C7 Tester names

- Suwat Inkaew 610610521

- Nadtaphong Jandaboot 630610743

## 5.1  Scope

Our software (Crowd Quizmaker) will be a cross-platform (Windows, MacOS, iOS, Android) web application that lecturer (or quiz admin) can create a quiz topic and students (or anyone - as quiz contributor) can contribute to a quiz - create, answer and review. Any users can be quizzes admin and/or quiz contributor and quizzes topic doesn't need to be academic purpose. Quizzes admin can specify what can quizzes contributor do.

## 5.2  Test scope

- Item to be tested

    - Desktop UI
    - Sign up function
    - Log in function
    - Create Quiz Topic function
    - Create Quiz (multiple choice format) function
    - Setting allowance of Quiz function
    - Manage Quiz function
    - Delete Quiz function
    - Attempt Quiz function
    - Log out function

- Item **NOT** to be tested

    - Cross platform UI
    - Review Quiz function
    - Create Quiz (Subjective format) function
    - Multiple correct choice function
    - Quiz Info show function

## 5.3  Test exit criteria

- Login and sign up authentication are working correctly

- UI and UX are satisfying the user

- Completing all specified types of testing with minimum number of errors, failures etc.

- Achieving a specified number of successful test cases

## 5.4 Test overview

- Unit Testing
- System Testing
- Integration Testing
- Acceptance Testing

## 5.5 Test types

- Unit Testing
    - Test techniques : White Box Testing
    - Description : Each unit tested by the developer to test thatunit working correctly.
    - Tester : Developer
- System Testing
    - Test techniques : Black Box Testing
    - Description : Done by the testing team by setting input randomly to check the return is according to requirement.
    - Tester : Testing Team
- Integration Testing
    - Test techniques : Big-bang testing
    - Description : Done by the testing team by runningapplications and trying all functions that integrate.
    - Tester : Testing Team
- Acceptance Testing
    - Test techniques : Alpha Testing, Beta Testing
    - Description :
        * Alpha testing is done by testing team by run a real-time scenario of quiz contributor and student
        * Beta testing done by TA
    - Tester : Testing Team, TA

## 5.6 Test result

- Unit Testing
    - Found some bugs and the developer already fixed it.
- System Testing
    - No holes detected, Already fix all holes.
- Integration Testing
    - All linked modules work normally.
- Acceptance Testing
    - Alpha test application works almost normally but still has some bugs in some functions.
    - Beta test application working normally.

## 5.7  Recommendation

Base on the results of the testing effort, the following recommendation are provided

- Tester team should provide more testing techniques and more tests to verify the quality and reliability of the software.

- Tester team should provide more clear information to the developer team. This can be improved by providing more meetings or making a minor test document.

## 5.8  Summary

The software testing effort for Crowd Quizmaker has been completed and the results are summarized in this report.

The overall results of testing indicate that the software performs well. However, several defects were identified during testing, which have been documented and reported to the developer to fix the problem.

The testing process was conducted using only manual testing techniques, including unit testing, system testing, integration testing and acceptance testing.

The testing team encountered several challenges during the testing process, including lack of communication, diversity in the testing environment and inadequate testing, which affected the testing effort.

Overall, the testing effort was successful in identifying defects and areas for improvement in the software.

# 6 A6 - Sprint #2

# Crowd Quizmaker (CQ)

Presented by 0N3 N16H7 PR0J3C7

**Stakeholders**
- Prof. Sakgasit Ramingwong
- Prof. Kampol Woradit

Members & Responsibility

**Chayanon Pitak**

as     Project Manager, System Analyst and Developer

# Responsibilities

- as Project Manager
  - Manage  overall of the project. Set the project up.
  - Distribute responsibility to members and plan the project.
  - Control the quality of the documents.
- as System Analyst
  - Discuss with stakeholders about the project.
  - Design the project according to the discussions.
- as Developer
  - Implement the front-end of the workspace category.

# Kritsanaphong Tepweerakul

as Business Analyst and Developer

## Responsibilities

- as Business Analyst
  - Discuss with stakeholders about the project.
  - Summarize the discussions for System analyst.
- as Developer
  - Implement authentication systems.
  - Implement the front-end of the contribution category.

# Kitpisan Tanngan

as Designer and Developer

## Responsibilities

- as Designer
  - Design the schema of the database and API.
- as Developer
  - Implement database and API schema.

**Woranut Kitchakan**

as      Designer

## Responsibilities

- as Designer
  - Design User Interface and User Experience of the software.

**Nadtaphong Jandaboot**

as Designer and Tester

# Responsibilities

- as Designer
  - Design User Interface and User Experience of the software.
- as Tester
  - Test the software in overall to ensure the quality of the software.
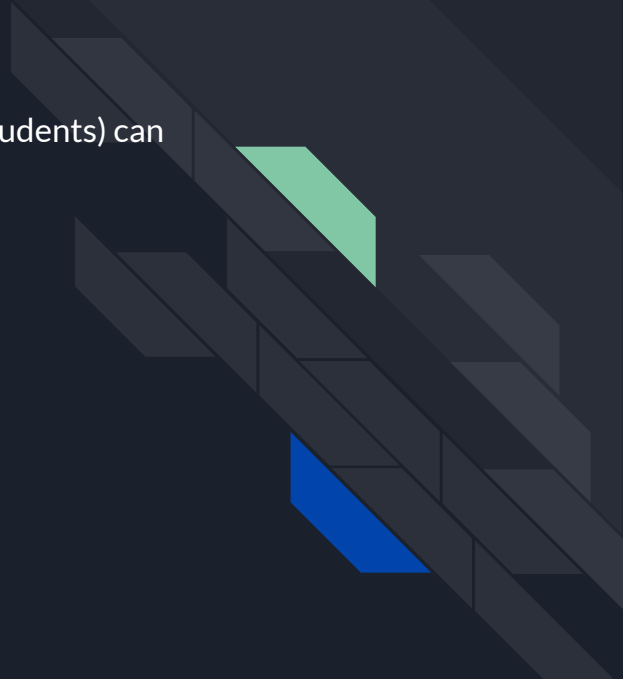
**Suwat Inkaew**

as     Tester

## Responsibilities

- as Tester
  - Test the software in overall to ensure the quality of the software.

# Introduction to Our Project

- Web application
- Online Quiz platform that user (either lecturer or students) can
  - Create, Edit and Delete quiz
  - Attempting the quiz created by another user

# Scope
## (Sprint #2)

Stakeholders are able to know the complete concept of web application including more details on Sprint#1 (Basic Authentication, wireframe etc.),

The software can communicate between server and client through API
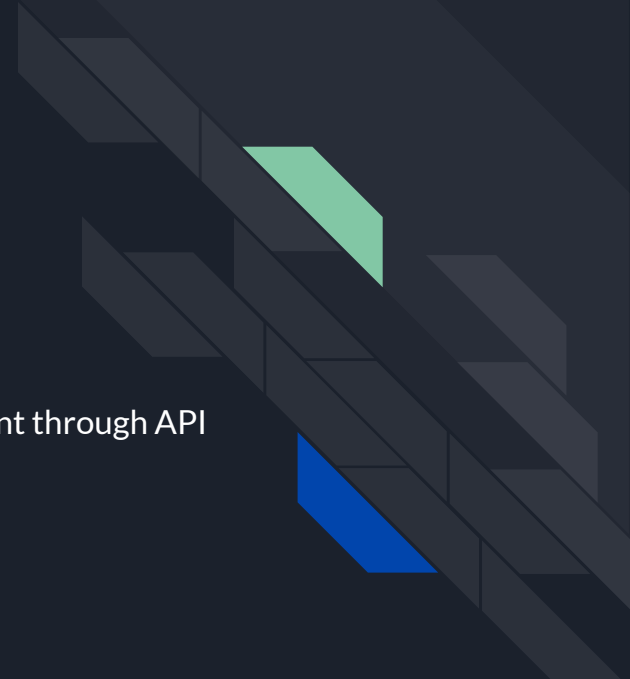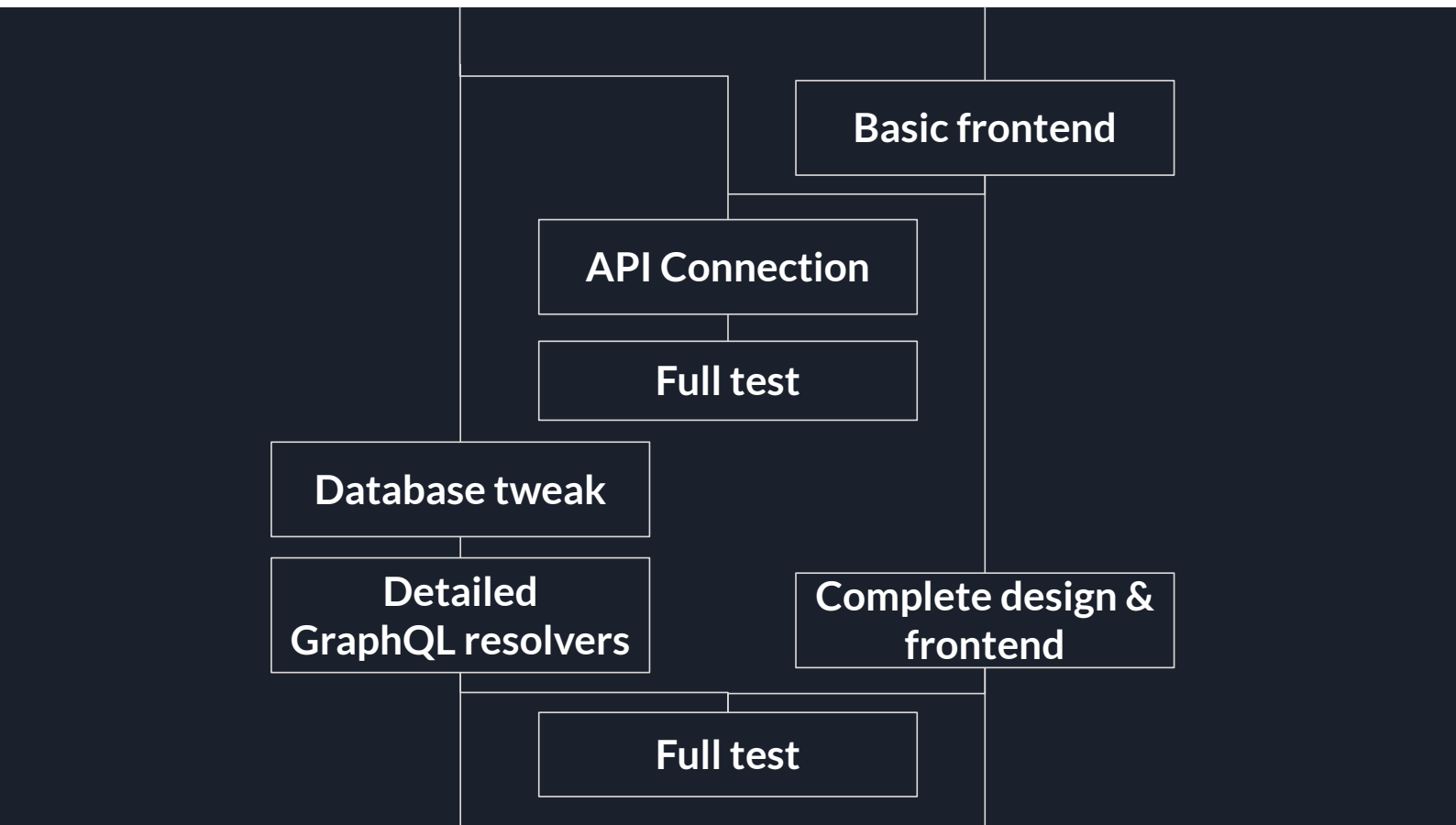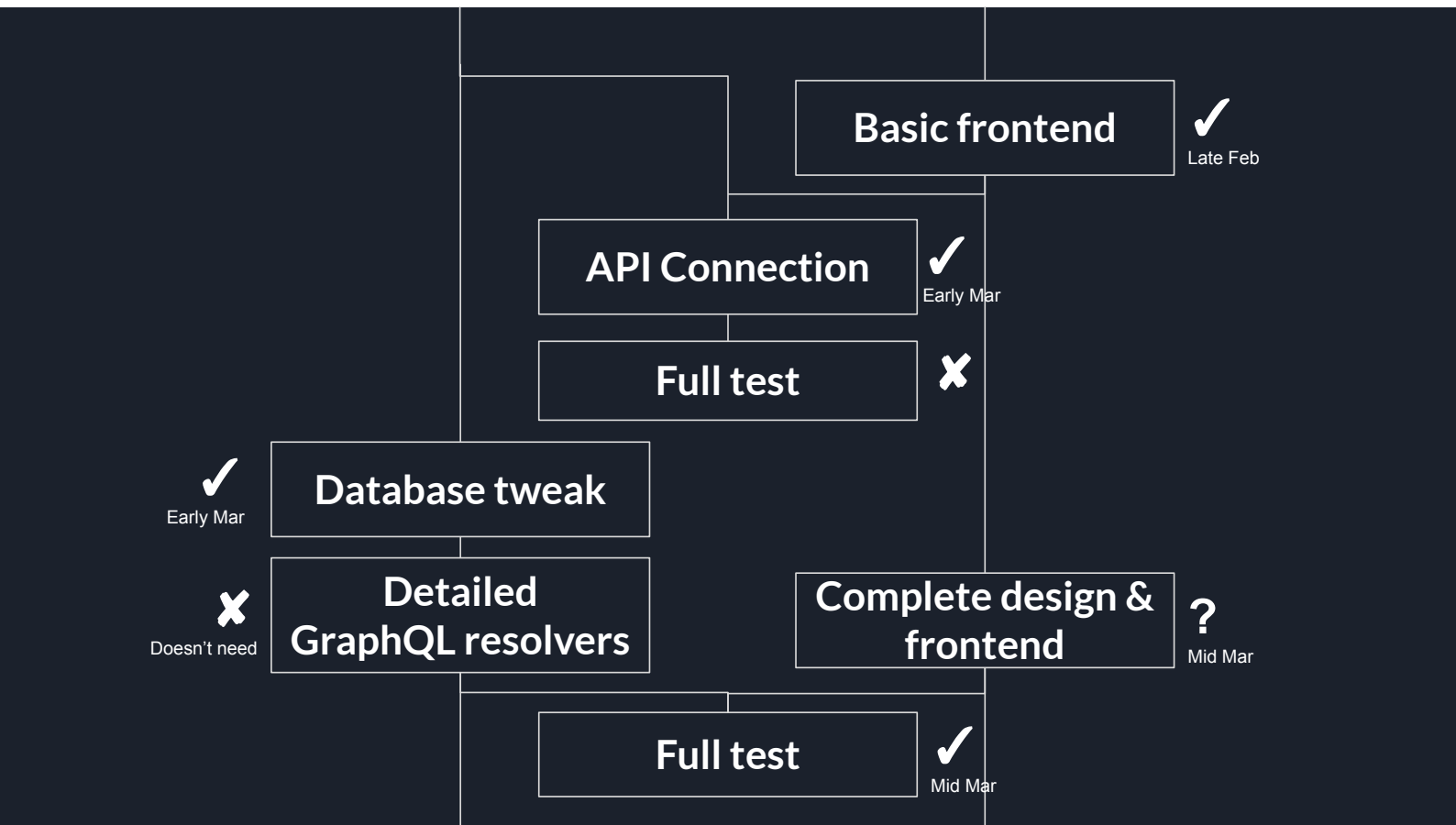
# Features
(Sprint #2)

- Contribute features
    - Create question
    - View past contributed quiz

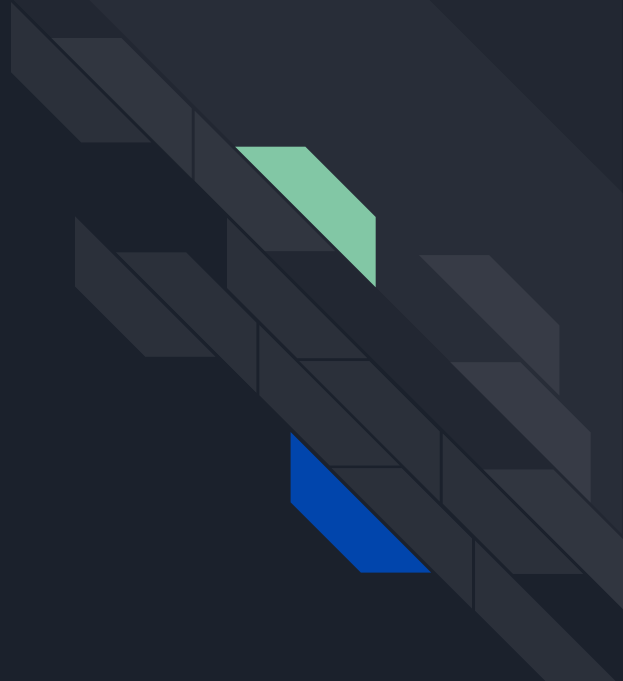- Quiz workspace features
    - Edit, manage and delete quiz

Retrospective

Basic frontend

API Connection

Full test

Database tweak

Detailed
GraphQL resolvers

Complete design &
frontend

Full test

**Basic frontend** ✓
Late Feb

**API Connection** ✓
Early Mar

**Full test** ✗

✓ **Database tweak**
Early Mar

✗ **Detailed GraphQL resolvers**
Doesn't need

**Complete design & frontend** ?
Mid Mar

**Full test** ✓
Mid Mar

# What wents well?

- Everyone participate 🎊
- Project (almost) goes as planned.

# What wents **NOT** well?

- No exact date on each backlog.
- Works until L457 N16H7 before sprint end.
- No proper (user) error handling.
- DEMO BUG ON PRESENTATION!!!

# What should we improve?

- Mark the deadline along with the backlog.
  and should have gap period before the final deadline in case of
  emergency happened
- Strict more on the progress.
- Let tester test what developer done more.
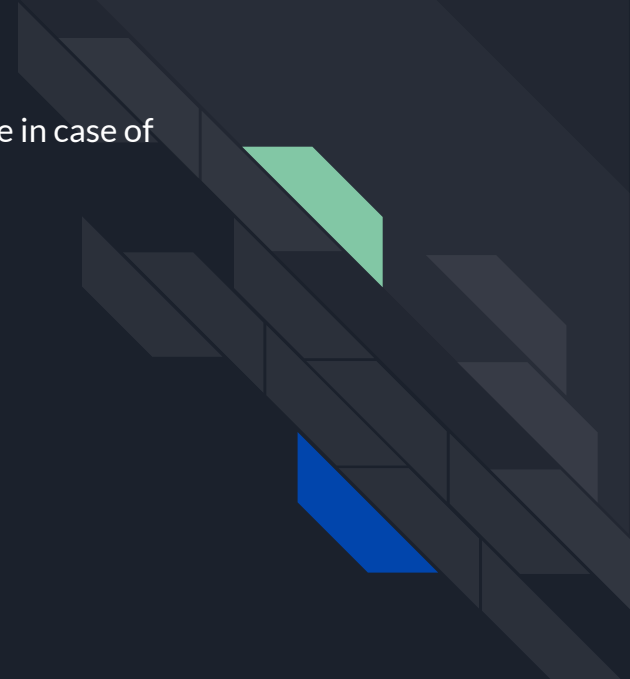
# VIDEO LINKS

https://o365cmu.sharepoint.com/sites/ProjectComputerArchitecture7/Shared%20Documents/261361%20-%200N3%20N16H7%20PR0J3C7/Recordings/A6-20230323_193422-Meeting%20Recording.mp4?web=1

# 7 Appendix

## 7.1 Source codes

Source code can be found at Github (https://github.com/ChayanonPitak/crowd-quizmaker/tree/main/source)

## 7.2 Acronyms and Abbreviations

- **API**: Application Programming Interface
- **LMS**: Learning Management System
- **GB**: Gigabyte
- **HTTP**: Hypertext Transfer Protocol
- **TB**: Terabyte