

Computer Engineering Essentials  
Report

UNO

Presented By

6530249621	Pavee Jeungtanasirikul
6530428621	Aunyaput Nitijarasrat
6632134521	Piyongkul Rardyota
6632135121	Poonyapat Jankrajang

2110221 Semester 2/2023

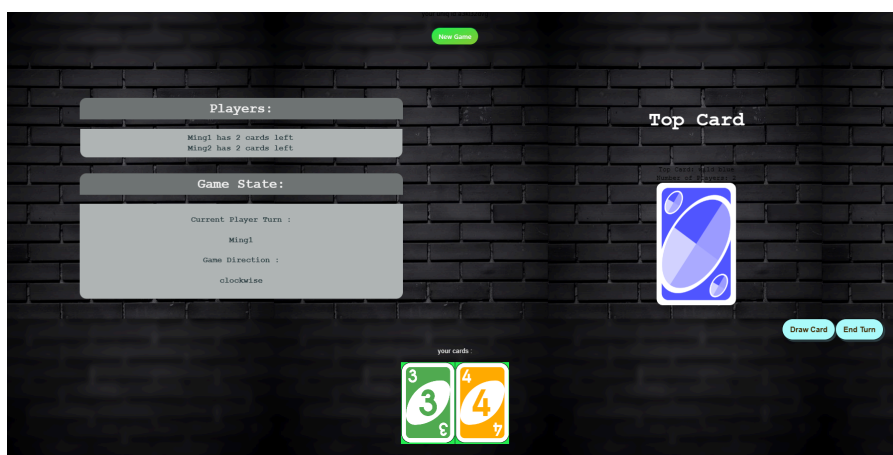
Department of Computer Engineering,  
Faculty of Engineering, Chulalongkorn University

## สารบัญ

Project Name	2
URL และวิธีใช้	3
Discussion regarding basic requirements	6
Discussion regarding challenging requirements	11
Project plan and actual execution	12j

# UNO

โปรเจกเกม uno เป็นเกมที่จะมีผู้เล่นอย่างน้อย 2 คนมาแข่งกันลงการ์ดตามกฎกติกาของเกม โดยผู้เล่นจะต้องลงการ์ดที่มีสีเหมือนกันกับการ์ดตรงกลาง หรือการ์ดที่มีตัวเลขเหมือนกันกับการ์ดบนสุดของกองกลาง และจะมีการ์ดที่มี Effect อยู่ด้วยโดยเกมนี้จะจบลงก็ต่อเมื่อผู้เล่นคนใดคนหนึ่งการ์ดหมดมือ จะถือว่าคนนั้นเป็นผู้ชนะในเกมนี้



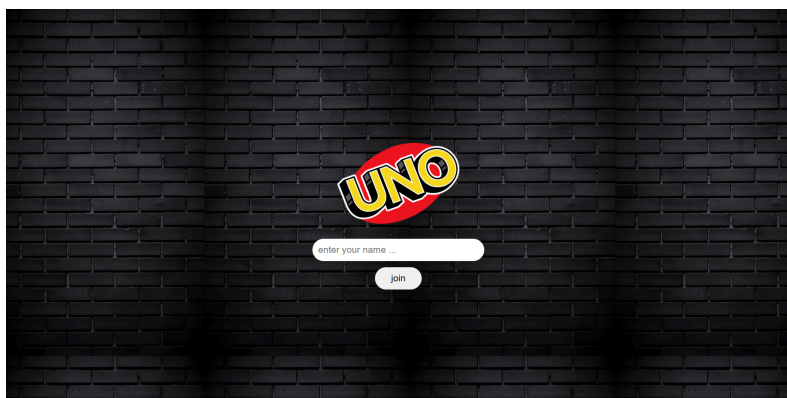
# URL and How to use the application

## URL

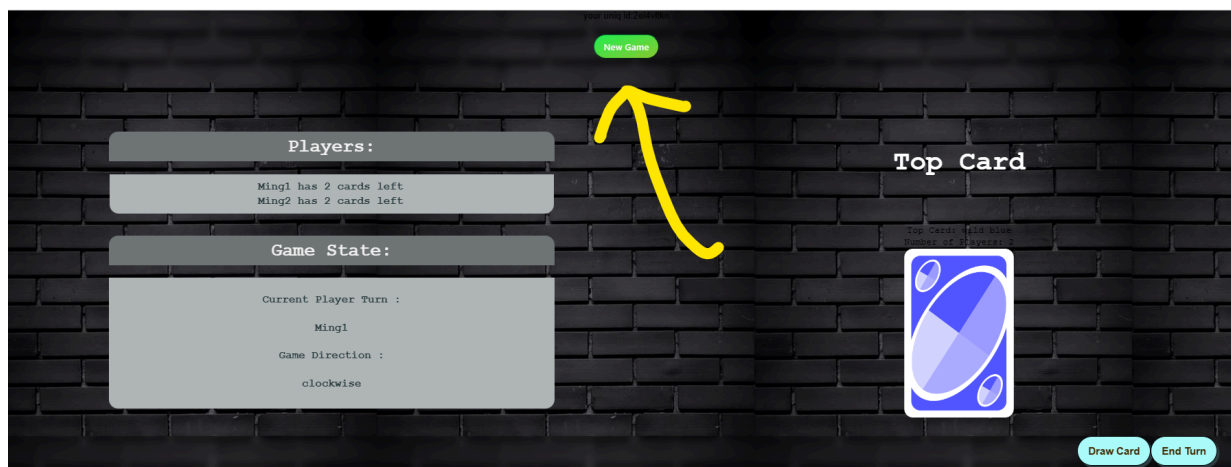
<http://18.213.250.212:3000/>

## วิธีใช้งาน

1. ใส่ชื่อผู้เล่นและกด join เพื่อเข้าร่วม



2. ให้ผู้เล่นคนใดคนหนึ่งกด New Game เพื่อที่เกมจะทำการแจกการ์ดให้กับทุกคน



3. ในแต่ละ turn ผู้เล่นจะสามารถทำได้ 1 อย่างนั้นคือ

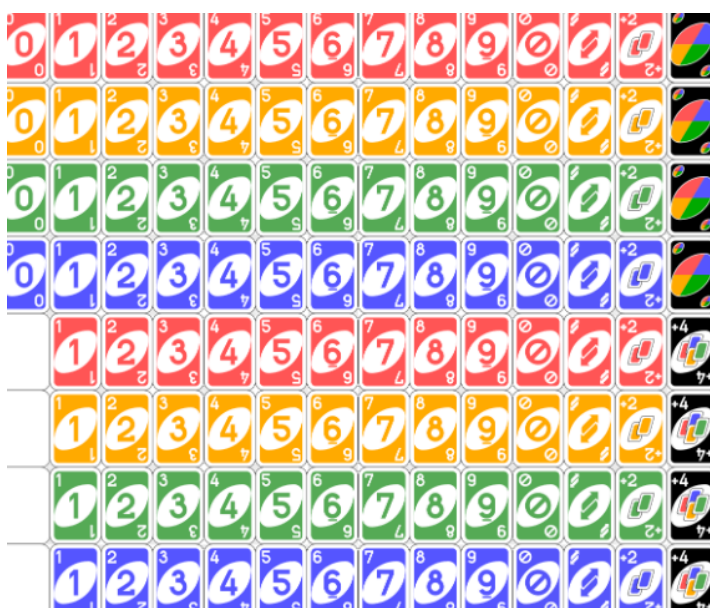
- Draw card
- Play card

และเมื่อจบเทิร์นจะต้องทำการกด End turn เพื่อให้ผู้เล่นคนต่อไปได้เล่น

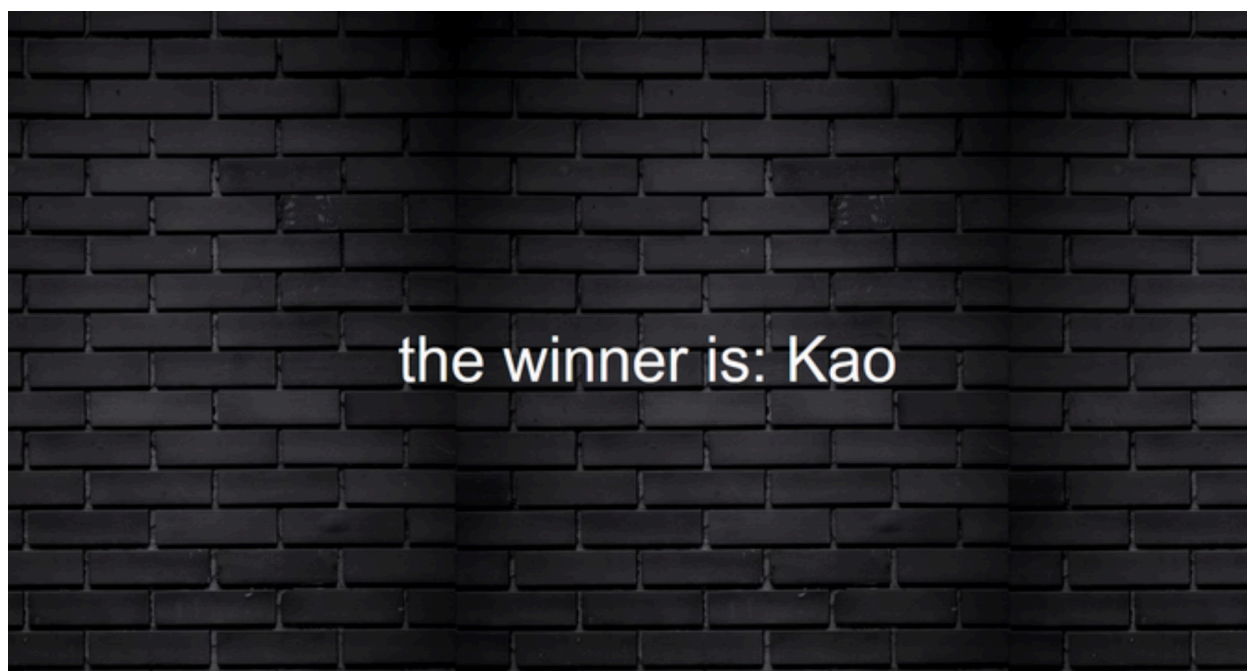
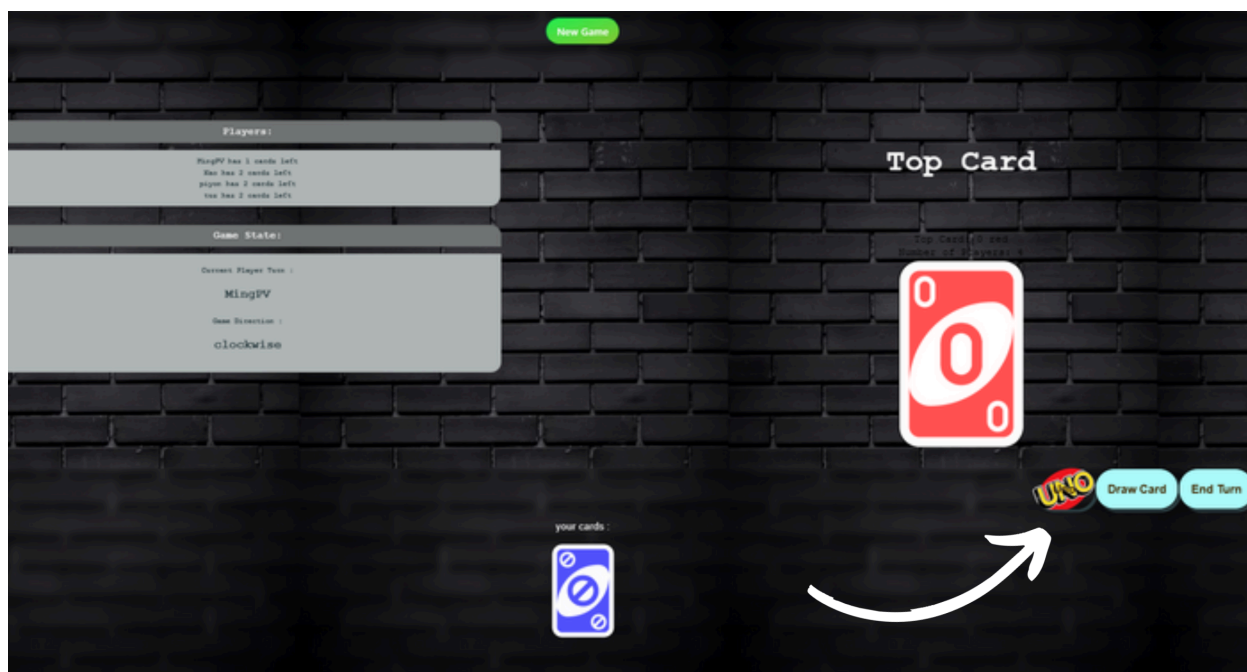


4. การ์ดจะมีทั้งหมด 2 ประเภทนั้นคือ Number card และ Effect card

- Number card มี 4 สีและแต่ละสีมีตัวเลข 0-9
- Effect card มี 4 ประเภท คือ +2 , +4 , wild card , skip card , reverse card



5.เมื่อผู้เล่นคนใดคนหนึ่งเหลือการ์ด 1 ใบจะต้องรีบกด uno ก่อนที่จะลงการ์ดใบสุดท้ายหากกดไม่ทันก็ต้องเล่นเกมต่อไป แต่หากกดทันและลงการ์ดใบสุดท้ายได้สำเร็จก็จะเป็นผู้ชนะในเกมนั้น



# Discussion regarding basic requirements

frontend ใช้ HTML CSS JavaScript

Backend ใช้ NodeJs ExpressJs

Database ใช้ Mongo

ไม่ได้มีการใช้ plugin, library, frameworkใดๆ

## Project Structure

- backend
  - node\_modules
  - src
    - config
      - db.js
    - controllers
      - cardController.js
      - gameController.js
      - playerController.js
      - tableController.js
    - models
      - cardModel.js
      - gameModel.js
      - playerModel.js
      - tableModel.js
    - routes
      - cardRoute.js
      - gameRoute.js
      - playerRoute.js
      - tableRoute.js
    - app.js
    - server.js
  - .env
  - .nvmrc
  - package-lock.json

- package.json
- frontend
  - node\_modules
  - public
    - scripts
      - assets
      - api.js
      - config.js
      - game.js
      - main.js
      - player.js
      - table.js
    - index.html
    - style.css
- .env
- package-lock.json
- package.json
- server.js



API List		
No.	Endpoint	Detail
1.1	GET /cards	การ์ดทั้งหมดในเกม
1.2	POST /cards	สร้างการ์ด
1.3	DELETE /cards/id	ลบการ์ด
1.4	POST /cards/update	แก้ไขข้อมูลการ์ด
2.1	POST /games	ใช้เริ่มเกม
2.2	GET /games/get	เรียกใช้ข้อมูลของเกม
2.3	GET /games/getRnd	จับการ์ดสุ่มจากเตี๊
2.4	POST /games/update	อัปเดตข้อมูลของเกม
2.5	GET /games/subscribeToUpdates	ใช้ SSE
2.6	POST /games/end	จบเกม
3.1	GET /players	ผู้เล่นทั้งหมดในเกม
3.2	POST /players	สร้างผู้เล่น
3.3	DELETE /players/id	ลบผู้เล่น
3.4	PUT /players/id/tmpcards	อัปเดตการ์ดในมือผู้เล่น
4.1	GET /tables	การ์ดใบบนสุด
4.2	POST /tables/card	สร้างการ์ด
4.3	DELETE /tables/id	ลบการ์ด

## Models

	playerModel
	unique (String)  Name (String)  Cards (Array)

	cardModel
	unique (String)  playername (String)  playerid (String)  value (String)  color (String)

	tableModel
	unique (String)  playername (String)  playerid (String)  value (String)  color (String)

	gameModel
	gameState (Boolean)
	gameDeck (Array)
	usedDeck (Array)
	gameDirection (Number)
	playerTurn (Number)
	skipFlags (Array)
	isPlayed (Boolean)
	isDraw (Boolean)
	drawId (String)
	pressedTime (Array)
	isPress (Boolean)
	isSkip (Boolean)

## Discussion regarding challenging requirements

### 1. Real-time with SSE

มีการทำreal-time ในการอัปเดตหน้าเว็บเมื่อผู้เล่นแต่ละคนลงไฟหรือจั่วไพ่ ให้อัปเดตอยู่ตลอดเวลาเหมือนเล่นจริงๆ

### 2. Responsive








































มีการตกแต่งหน้าเว็บให้เหมาะสมกับขนาดหน้าจออุปกรณ์ที่ผู้เล่นใช้ในหลากหลายขนาด เช่น โทรศัพท์ แท็บเล็ต และขนาดเดสท็อป

## Project plan and actual execution

### 1. Project plan

- 12-13 April : เริ่มวางแผน เลือกหัวข้อ และออกแบบตัวเกมคร่าวๆ
- 13-14 April : สร้างตัวเก็บข้อมูลพื้นฐานพร้อมกับฟังก์ชันพื้นฐาน  
(สร้าง ลบ อัปเดต ข้อมูล)
- 15-16 April : Game logic
- 17-18 April : ออกแบบหน้าเว็บและพัฒนาตัวเกมเพิ่มเติม
- 19-22 April : ทำให้สามารถเล่นพร้อมกันได้ และพัฒนาหน้าเว็บ พร้อมกับทดสอบ
- 23 April ทดสอบตัวเกมปรับปรุงตัวเกมรอบสุดท้ายและdeploy

Gantt Chart

Gantt Chart													
List	12 Apr	13 Apr	14 Apr	15 Apr	16 Apr	17 Apr	18 Apr	19 Apr	20 Apr	21 Apr	22 Apr	23 Apr	24 Apr
Plan													
Design													
Front-end Dev													
Back-end Dev													
Integration													
UAT Test													
Deploying													
Review & Present													

## 2. Project Execute

### 2.1 Front-end Development

- 2.1.1 สร้างหน้าเว็บที่เอาไว้ทดสอบ
- 2.1.2 สร้างปุ่มต่างๆกับแสดงข้อมูลต่างๆ โดยที่ยังไม่ได้จัดหน้าเว็บและ  
ตกแต่งหน้าเว็บ เนื่องจากเอาไว้ทดสอบอย่างเดียว
- 2.1.3 พัฒนา game logic ที่เกี่ยวข้องกับ frontend
- 2.1.4 ออกแบบหน้าเว็บ
- 2.1.5 จัดหน้าเว็บและสร้างปุ่มให้ครบ
- 2.1.6 ตกแต่งและปรับปรุงให้สมบูรณ์

### 2.2 Back-end Development

- 2.2.1 เริ่มสร้าง model หลักๆมาเก็บข้อมูล (playerModel ,  
cardModel, tableModel ...)
- 2.2.2 สร้างฟังก์ชันให้สามารถ สร้างข้อมูล ลบข้อมูล อัปเดตข้อมูล
- 2.2.3 พัฒนา game logic ที่เกี่ยวข้องกับ backend
- 2.2.4 ทำ real time feature โดยใช้ SSE
- 2.2.5 ปรับปรุงและแก้ไขให้สมบูรณ์