ข้อสอบกลางภาค วิชา 10301347 วิศวกรรมการพัฒนาปฏิบัติการบนคลาวด์

สอบวันที่ 20 ม.ค. 69 ชั้น 6 ตึกคณะวิทยาศาสตร์ 60 ปี ห้องปฏิบัติการคอมพิวเตอร์ 2

เวลาสอบ 13.00-16.00 น. ข้อสอบคะแนนรวม 100 คะแนน ประกอบด้วย

1) ขั้นตอนการปฏิบัติ: สร้าง Branch ให้สอดคล้อง (30 คะแนน)

2) ขั้นตอนการปฏิบัติ: Commit ทุกขั้นตอน + ก าหนด Message (40 คะแนน)

3) ขั้นตอนการปฏิบัติ: Push แต่ละ Branch ขึ้น GitHub (30 คะแนน)

ให้นักศึกษาส่ง GitHub link เพื่อเป็นค าตอบของการสอบ

1) ขั้นตอนการปฏิบัติ: สร้าง Branch ให้สอดคล้อง (30 คะแนน)

เงื่อนไขหลัก: ทุก branch ต้องสร้างจาก main (หรือ master) ล่าสุด และชื่อ branch ต้อง "สื่อความหมายตรงงาน" รายการ Branch ที่ต้องสร้าง (ให้ท าตามล าดับ)

ขั้นตอน	สิ่งทที่ า	Branch ทตี่ ้องสร้าง	คะแนน
B1	สร้างโครง backend + ติดตั้ง deps	feature/backend-init	6
B2	เขียน server.js + /api/demo + logs + error handling + CORS/dotenv	feature/backend-api demo	6
B3	ติดตั้ง axios + แก้ IndexPage.vue เรียก API + เพิ่ม .env (frontend)	feature/frontend-axios integration	6
B4	เพิ่ม .gitignore ให้เหมาะสม (เช่น ignore backend/logs/ และไฟล์ลับ)	chore/gitignore-update	4
B5	สร้าง backend/Dockerfile + .dockerignore	chore/dockerize-backend	4
B6	สร้าง/อัปเดต docker-compose.yml ให้รัน full-stack + network + volume + healthcheck	chore/compose-fullstack	4
รวม			30

เกณฑ์ให้คะแนน (Branch)

• ได้เต็ม: สร้างครบทุก branch, ชื่อถูกต้อง/สื่อความหมาย, checkout ไปทำงานใน branch นั้นจริง • หัก: ชื่อไม่สื่อ/สะกดผิด/ใช้ branch เดียวทำทุกอย่าง/สร้างจาก branch อื่นที่ไม่ใช่ main

2) ขั้นตอนการปฏิบัติ: Commit ทุกขั้นตอน + กำหนด Message (40 คะแนน)

แนวคิด: 1 branch ควรมีอย่างน้อย 1 commit ที่ "จบงานชัดเจน" (ถ้ามีหลาย commit ได้ แต่ต้อง message ดี) ตาราง Commit ที่ "ต้องมี" (แนะนำอย่างน้อย 1 commit/branch)

Branch ไฟล์งานที่คาดหวัง	Commit message ที่ "กำหนดให้ใช้"	คะแนน
feature/backend-init	backend/package.json	feat: initialize express backend with core dependencies	7
feature/backend api-demo	เพิ่ม frontend/.env *(ต้องไม่ commit .env ถ้ากำหนดให้ ignore)*	integrate quasar frontend with backend api using axios	8
feature/frontend axios-integration	อัปเดต .gitignore (root) เช่น ignore backend/logs/, .env, node_modules	chore: update gitignore for fullstack project	6
chore/gitignore update	backend/Dockerfile, backend/.dockerignore, healthcheck ตามโจทย์	chore: dockerize backend with multi stage build and healthcheck	6
chore/dockerize backend	docker-compose.yml มี 2 services, network, volume logs, env vars	chore: add docker compose for fullstack with network and volumes	6
chore/compose fullstack	backend/server.js, สร้าง logs folder, endpoint /api/demo, error handling	feat: add /api/demo endpoint with logging, cors, and error handling	7
feat:	frontend ติดตั้ง axios, แก้ IndexPage.vue เรียก API,		

รวม 40 เกณฑ์ตรวจส าคัญ (Commit)

• message **ตรงตาม "ที่ก าหนดให้ใช้" (หรือความหมายเทียบเท่า + รูปแบบ** Conventional Commits) • commit **มีไฟล์จริงตามงานในตาราง (ไม่ใช่** commit **เปล่า)**

• **ไม่** commit **ไฟล์ลับ เช่น** .env **(ถ้าอาจารย์ก าหนดให้** ignore**) และไม่เอา** backend/logs/ **ขึ้น** git • **แนะน า:** git status **สะอาดหลัง** commit

3) ขั้นตอนการปฏิบัติ: Push แต่ละ Branch ขึ้น GitHub (30 คะแนน)

เงื่อนไขหลัก: ทุก branch **ต้องปรากฏบน** GitHub **และตรวจสอบได้จากหน้า** branches **หรือ** PR **เกณฑ์การ** Push **(ให้ท าครบทุก** branch)

งาน	วิธีตรวจ	คะแนน
P1: PUSH FEATURE/BACKEND-INIT	**เห็น** branch **บน** GitHub + commit **ล่าสุดตรง**	5
P2: PUSH FEATURE/BACKEND-API-DEMO	**เห็น** branch **บน** GitHub + commit **ล่าสุดตรง**	5
P3: PUSH FEATURE/FRONTEND-AXIOS INTEGRATION	**เห็น** branch **บน** GitHub + commit **ล่าสุดตรง**	5
P4: PUSH CHORE/GITIGNORE-UPDATE	**เห็น** branch **บน** GitHub + commit **ล่าสุดตรง**	5
P5: PUSH CHORE/DOCKERIZE-BACKEND	**เห็น** branch **บน** GitHub + commit **ล่าสุดตรง**	5
P6: PUSH CHORE/COMPOSE-FULLSTACK	**เห็น** branch **บน** GitHub + commit **ล่าสุดตรง**	5

รวม 30 ค าสั่ง push **ตัวอย่าง (นักศึกษาต้องท าในแต่ละ** branch)

git push -u origin <branch-name>

หลักฐานที่แนะน าให้ส่ง

- **ลิงก์หน้า** GitHub repo (**แท็บ** Branches **หรือ** PR)
- **หรือ** screenshot **รายชื่อ** branch + commit message **ล่าสุดของแต่ละ** branch

Step 0: เตรียมโปรเจกต์/อัปเดต main (ก่อนเริ่ม)

ท าบน main

- **ค าสั่ง**
  - git checkout main
  - git pull origin main

Step 1: สร้าง Backend (โครงสร้าง + ติดตั้งแพก็เกจ)

Branch: feature/backend-init

สิ่งที่ต้องทา (ตามโจทย์)

- **สร้างโฟลเดอร์**backend/
- npm init -y
- npm install express cors dotenv

ไฟล์/โฟลเดอรท์ คี่ วรมี

- backend/package.json
- backend/package-lock.json

Commit message (ก าหนดให้ใช้)

- feat: initialize express backend with core dependencies

Push

- git push -u origin feature/backend-init

Step 2: สร้าง API Backend + logging + error handling

Branch: feature/backend-api-demo

สิ่งทตี่ ้องทา

- **สร้าง** backend/server.js
- **เพิ่ม** /api/demo return JSON (git/docker info) •

สร้าง backend/logs/ **ถ้ายังไม่มี และเขียน** log

access.log • **เปิดใช้**cors, express.json()

- **เพิ่ม** error handling middleware

ไฟล์/โฟลเดอรท์ คี่ วรมี

- backend/server.js
- backend/logs/ (**แต่** ***ห้าม*** commit logs **ขึ้น** git)

Commit message

- feat: add /api/demo endpoint with logging, cors, and error handling

Push

- git push -u origin feature/backend-api-demo

Step 3: อัปเดต Frontend (Quasar) ให้เรียก API ด้วย Axios + env

Branch: feature/frontend-axios-integration

สิ่งทตี่ ้องทา

- frontend/ **ติดตั้ง** axios
- **แก้**frontend/src/pages/IndexPage.vue **ให้เรียก** API **จาก** VITE_API_URL +

'/api/demo' • **สร้าง** frontend/.env **ส าหรับ** dev: VITE_API_URL=http://localhost:3000 ○

หมายเหตุ: โดยทั่วไป .env **ควรถูก** ignore (**แนะน าท า** .env.example **ส า**

หรับส่งขึ้น git) **ไฟลท์ คี่ วรมี**

- frontend/src/pages/IndexPage.vue
- frontend/.env (**ไม่ควร** commit) **หรือ**frontend/.env.example (**ควร** commit)

Commit message

- feat: integrate quasar frontend with backend api using axios

Push

- git push -u origin feature/frontend-axios-integration

Step 4: ปรับ .gitignore ให้เหมาะกับ full-stack

Branch: chore/gitignore-update

สิ่งที่ต้องทำ

- อัปเดต .gitignore ที่ root ให้ ignore อย่างน้อย:
  - backend/logs/
  - **/.env (หรือเฉพาะ frontend/.env, backend/.env)
  - node_modules/

ไฟล์

- .gitignore

Commit message

- chore: update gitignore for fullstack project

Push

- git push -u origin chore/gitignore-update

Step 5: Dockerize Backend (Dockerfile + .dockerignore + healthcheck)

Branch: chore/dockerize-backend

สิ่งที่ต้องทำ

- สร้าง backend/Dockerfile แบบ multi-stage
- สร้าง backend/.dockerignore
- เพิ่ม healthcheck (เช็ค /api/demo)
- สร้าง logs dir ใน container

ไฟล์

- backend/Dockerfile
- backend/.dockerignore

Commit message

- chore: dockerize backend with multi-stage build and healthcheck

Push

- git push -u origin chore/dockerize-backend

Step 6: Docker Compose รัน Full-stack (frontend + backend + network + volume)

Branch: chore/compose-fullstack

สิ่งทตี้องทา

- สร้าง/อัปเดต docker-compose.yml ที่ root
- มี 2 services: frontend, backend
- ตั้ง VITE_API_URL=http://backend:3000 ใน frontend
- ตั้ง network app-network
- ตั้ง volume ./backend/logs:/app/logs เพื่อ persist logs
- มี healthcheck (ถ้าก าหนดไว้ใน compose)

ไฟล์

- docker-compose.yml

Commit message

- chore: add docker compose for fullstack with network and volumes

Push

- git push -u origin chore/compose-fullstack