Architectural Drivers Document

ชื่อระบบ : Tinder

ประเภทของระบบ : ☑ Social Media Platform ☐ E-Learning Platform ☐ Smart Home IOT ☐ Food Delivery System ☐ Hotel Booking System ☐ Healthcare/Clinic Management

ส่วนที่ 1 : System Overview

System Descriotion :

Tinder ซึ่งเป็นแอปพลิเคชันด้านการจับคู่ที่ได้รับความนิยมทั่วโลก ระบบของ Tinder ถูกออกแบบมาเพื่อช่วยให้ผู้ใช้สามารถค้นหาเพื่อนใหม่หรือคู่เดทได้อย่างง่าย รวดเร็ว และปลอดภัย โดยระบบมีโครงสร้างการทำงานหลักอยู่หลายส่วน

เริ่มจากระบบ User Account & Profile ผู้ใช้สามารถสมัครสมาชิกผ่านเบอร์โทรศัพท์ อีเมล หรือบัญชีโซเชียลมีเดีย จากนั้นจะสร้างโปรไฟล์ที่ประกอบด้วยรูปภาพ ความสนใจ และข้อมูลส่วนตัวต่างๆ เพื่อใช้เป็นพื้นฐานในการจับคู่

ต่อมาคือระบบที่สำคัญที่สุด คือ Recommendation Engine หรือระบบแนะนำคู่ ซึ่งจะนำข้อมูลตำแหน่งที่อยู่ ความสนใจ และพฤติกรรมการกด Like หรือปัดหน้าจอของผู้ใช้ มาวิเคราะห์เพื่อคัดเลือกโปรไฟล์ที่มีความเหมาะสมที่สุดให้ปรากฏบนหน้าจอ

เมื่อผู้ใช้เจอโปรไฟล์ที่สนใจ ก็สามารถ Swipe Right เพื่อแสดงความสนใจ หรือ Swipe Left เพื่อผ่านไป หากทั้งสองฝ่ายปัดขวาให้กัน ระบบจะถือว่า ‘Match’ และทำการเปิดหน้าต่างแชทให้ทั้งคู่สามารถพูดคุยกันได้ทันที พร้อมส่งการแจ้งเตือนแบบ Real-time

นอกจากนี้ ระบบยังมีการบริหารจัดการด้านความปลอดภัย เช่น การตรวจจับโปรไฟล์ปลอม การรายงานผู้ใช้ และมีฟีเจอร์เสริม เช่น Super Like, Boost และบริการสมาชิกแบบพรีเมียม เพื่อเพิ่มโอกาสในการเจอคู่ที่ตรงใจ

โดยรวมแล้ว Tinder เป็นระบบที่ผสานการใช้งานแบบง่ายดายเข้ากับเทคโนโลยี Machine Learning เพื่อแนะนำคู่ที่ตรงความสนใจของผู้ใช้ พร้อมสร้างประสบการณ์การใช้งานที่รวดเร็ว ปลอดภัย และเข้าถึงได้ทุกคน”

Target Users :

- บุคคลทั่วไปที่ต้องการค้นหาคู่เดท
- คนโสดที่ต้องการความสัมพันธ์จริงจัง
- ผู้ที่ต้องการเพื่อนใหม่หรือเครือข่ายสังคม
- คนทำงาน/วัยรุ่นที่ต้องการการเชื่อมต่อแบบรวดเร็ว
- ผู้ใช้ที่สนใจแลกเปลี่ยนคอนเทนต์ไลฟ์สไตล์

Key Features :

1. Swipe Matching System (ปัดซ้าย/ขวา) :
   ระบบหลักที่ทำให้ Tinder แตกต่าง — ใช้งานง่าย ตัดสินใจเร็ว และลดความยุ่งยากในการเลือกคู่
2. Mutual Match (จับคู่เมื่อทั้งสองฝ่ายสนใจ) :
   ระบบจะเปิดแชทเฉพาะเมื่อทั้งสองคนปัดขวาให้กัน ช่วยลดการรบกวนและเพิ่มความเป็นส่วนตัว
3. Smart Recommendation & Algorithm : ใช้ Machine Learning วิเคราะห์ความสนใจ พฤติกรรมการปัด และตำแหน่ง เพื่อแนะนำโปรไฟล์ที่เหมาะสมที่สุด
4. Location-based Matching (จับคู่ตามระยะทาง) : ใช้ GPS เพื่อค้นหาผู้ใช้ที่อยู่ใกล้ ทำให้เจอคนที่สามารถพบเจอกันได้จริง
5. User Profile & Interests : แสดงรูปภาพ, Bio, Interests, Spotify Top Artists, Instagram Link เพื่อสร้างตัวตนที่แท้จริง
6. Real-time Chat System : เมื่อ Match แล้วสามารถส่งข้อความ รูปภาพ GIF Emoji ได้แบบ Real-time พร้อม Push Notification
7. Super Like / Boost / Super Boost : ฟีเจอร์เพิ่มโอกาสการถูกเห็นมากขึ้น เหมาะสำหรับผู้ใช้ที่ต้องการสร้างความโดดเด่น
8. Passport Mode (ค้นหาคนในประเทศอื่น) : ผู้ใช้สามารถเปลี่ยนตำแหน่งเพื่อดูโปรไฟล์จากพื้นที่อื่น เหมาะสำหรับนักเดินทางและชาวต่างชาติ

9. Safety & Moderation Tools : ระบบ AI ตรวจจับโปรไฟล์ปลอม, การรายงานผู้ใช้, การบล็อก, การยืนยันตัวตนด้วยรูปถ่าย (Photo Verification)
10. Premium Subscriptions (Plus / Gold / Platinum) : ปลดล็อกฟีเจอร์พิเศษ เช่น Unlimited Likes, See Who Likes You, Rewind, Priority Likes เพื่อเพิ่มผลลัพธ์ในการจับคู่

ส่วนที่ 2 : Functional Requirements

User Management

FR-01 : ลงทะเบียนผู้ใช้ (User Registration)

ระบบต้องอนุญาตให้ผู้ใช้ลงทะเบียนบัญชีใหม่ได้โดยใช้

- เบอร์โทรศัพท์ หรืออีเมล
  และต้องมีการยืนยันตัวตน (เช่น OTP หรือ Email Verification) ก่อนใช้ระบบ

FR-02 : เข้าสู่ระบบ / ออกจากระบบ (Login / Logout)

ระบบต้องรองรับการเข้าสู่ระบบด้วยข้อมูลที่ลงทะเบียนไว้
และต้องสามารถออกจากระบบได้อย่างปลอดภัยทุกเวลา

FR-03 – จัดการโปรไฟล์ผู้ใช้ (Edit Profile)

ระบบต้องอนุญาตให้ผู้ใช้แก้ไขข้อมูลโปรไฟล์ของตนเองได้ เช่น

- รูปโปรไฟล์
- Bio (แนะนำตัว)
- อายุ เพศ ความสนใจ (Interests)

FR-04 : ตั้งค่าความเป็นส่วนตัว (Privacy Settings)

ระบบต้องให้ผู้ใช้สามารถตั้งค่าความเป็นส่วนตัวได้ เช่น

- ซ่อน/แสดงอายุ

- ซ่อน/แสดงระยะทาง
- เลือกเพศของผู้ใช้ที่ต้องการให้เห็นโปรไฟล์ตนเอง

FR-05 ปิด/ลบบัญชีผู้ใช้ (Deactivate / Delete Account)

ระบบต้องรองรับการปิดใช้งานชั่วคราว (Deactivate) และการลบบัญชีถาวร
โดยเมื่อผู้ใช้ลบบัญชีข้อมูลที่เกี่ยวข้องต้องถูกจัดการตามนโยบายระบบ

Core Features

FR-06 : แสดงรายการโปรไฟล์ที่แนะนำ (Candidate Listing) :

ระบบต้องแสดงโปรไฟล์ผู้ใช้อื่นตามเกณฑ์ที่กำหนด (เช่น ระยะทาง อายุ เพศ) เพื่อให้ผู้ใช้ทำการปัด (Swipe)

FR-07 : ปัดซ้าย (Dislike / Pass)

ระบบต้องรองรับการปัดซ้ายเพื่อระบุว่าไม่สนใจโปรไฟล์นั้น
และต้องไม่แสดงโปรไฟล์เดิมซ้ำในช่วงเวลาหนึ่ง (หรือไม่แสดงอีกเลยตามกติกา)

FR-08 : ปัดขวา (Like)

ระบบต้องรองรับการปัดขวาเพื่อระบุว่าสนใจโปรไฟล์นั้น และต้องบันทึกการกระทำเพื่อใช้ตรวจจับ Mutual Match

FR-09 : Super Like

ระบบต้องรองรับฟีเจอร์ Super Like โดยจำกัดจำนวนการใช้งานต่อวันตามสิทธิ์ของผู้ใช้แต่ละประเภท และให้ระบุสถานะพิเศษกับผู้ถูก Super Like

FR-10 : ตรวจจับการ Match (Mutual Match Detection)

เมื่อผู้ใช้สองคนปัดขวาให้กัน ระบบต้องตรวจจับว่าเกิด Match และสร้างความสัมพันธ์ (Match Record) ระหว่างผู้ใช้ทั้งสอง

FR-11 : เปิดห้องสนทนาเมื่อ Match (Open Chat on Match)

เมื่อเกิด Match ระบบต้องสร้างห้องสนทนาระหว่างผู้ใช้ทั้งสองโดยอัตโนมัติ
เพื่อให้สามารถเริ่มแชทได้ทันที

FR-12 : ส่งข้อความระหว่างคู่ Match (Messaging)

ระบบต้องรองรับการส่งข้อความระหว่างผู้ใช้ที่ Match กันแล้ว
โดยข้อความต้องถูกส่งและแสดงผลแบบใกล้เคียง Real-time

FR-13 : ยกเลิกการ Match (Unmatch)

ระบบต้องอนุญาตให้ผู้ใช้สามารถยกเลิก Match กับอีกฝ่ายได้
และเมื่อยกเลิกแล้วต้องไม่สามารถส่งข้อความหากันได้อีก

Notification

FR-14 : การแจ้งเตือนเมื่อเกิด Match ใหม่

ระบบต้องส่งการแจ้งเตือนให้ผู้ใช้เมื่อมี Match ใหม่เกิดขึ้น (เช่น Push Notification / In-app Notification)

FR-15 : การแจ้งเตือนเมื่อมีข้อความใหม่

ระบบต้องส่งการแจ้งเตือนให้ผู้ใช้เมื่อมีข้อความใหม่จากคู่ Match ที่ยังไม่ได้อ่าน

FR-16 : การตั้งค่าการแจ้งเตือน (Notification Settings)

ระบบต้องให้ผู้ใช้สามารถเปิด/ปิด หรือปรับรูปแบบการแจ้งเตือนได้ เช่น

- ปิดเสียงแจ้งเตือน
- ปิดเฉพาะบางประเภท (เช่น แจ้งเตือนข้อความอย่างเดียว ไม่แจ้งเตือน Match)

Reporting

FR-17 : รายงานโปรไฟล์ (Report Profile)

ระบบต้องให้ผู้ใช้สามารถรายงานโปรไฟล์ที่มีพฤติกรรมไม่เหมาะสม เช่น โปรไฟล์ปลอม การคุกคาม เนื้อหาไม่เหมาะสม โดยต้องมีฟอร์มให้เลือกสาเหตุการรายงาน

FR-18 : รายงานการสนทนา (Report Chat / Message)

ระบบต้องอนุญาตให้ผู้ใช้รายงานบทสนทนาเฉพาะรายการ หรือข้อความบางส่วนที่ไม่เหมาะสมได้ เพื่อให้ทีมงานตรวจสอบ

FR-19 : บล็อกผู้ใช้ (Block User)

ระบบต้องรองรับการบล็อกผู้ใช้รายอื่น เพื่อไม่ให้

- เห็นโปรไฟล์กัน
- ส่งข้อความหากันได้
- รวมถึงต้องไม่แนะนำโปรไฟล์ของผู้ที่ถูกบล็อกกลับมาอีก

FR-20 : การจัดการรายงานโดยผู้ดูแลระบบ (Admin Handling)

ระบบต้องมีส่วนของผู้ดูแล (Admin/Moderator) สำหรับ

- ดูรายการ Report ที่ส่งเข้ามา
- ตรวจสอบข้อมูลที่เกี่ยวข้อง (ประวัติการสนทนา/โปรไฟล์)
- ดำเนินการเตือน ระงับ หรือแบนบัญชีผู้ใช้ตามนโยบาย

ส่วนที่ 3 : Quality Attributes & Scenarios

QA-1 : Performance

Scenario: การโหลดโปรไฟล์และการปัด (Swipe) ต้องรวดเร็วแม้มีผู้ใช้งานจำนวนมาก

ส่วน	รายละเอียด
Source	ผู้ใช้ทั่วไปที่กำลังใช้งาน Tinder (Mobile App / Web)
Stimulus	ผู้ใช้ทำการปัดขวา หรือปัดซ้าย เพื่อโหลดโปรไฟล์ถัดไปอย่างต่อเนื่อง
Artifact	ระบบ Matching Engine, Recommendation Service, User Profile Service
Environment	ผู้ใช้ใช้งานในช่วงเวลา Peak Time (เช่น ช่วงค่ำ) มีผู้ใช้หลายล้านคนออนไลน์พร้อมกัน
Response	ระบบต้องโหลดโปรไฟล์ถัดไปทันทีโดยไม่เกิดอาการหน่วง พร้อมบันทึกข้อมูลการปัดอย่างถูกต้อง
Response Measure	เวลาแสดงโปรไฟล์ใหม่ ≤ 1 วินาที

คำอธิบายเพิ่มเติม :

ประสิทธิภาพ (Performance) เป็นหัวใจสำคัญของ Tinder เพราะลักษณะการใช้งานของระบบคือการ “ปัด” โปรไฟล์อย่างต่อเนื่อง ซึ่งต้องการการโหลดข้อมูลแบบรวดเร็วทันใจ หากการตอบสนองช้าเพียง 1–2 วินาทีจะทำให้ผู้ใช้รู้สึกว่าระบบไม่ลื่นไหลและเกิดความไม่พอใจอย่างมาก

ระบบของ Tinderจึงต้องออกแบบให้รองรับการประมวลผลปริมาณมาก เช่น การโหลดโปรไฟล์ การส่ง/บันทึกรายการปัด และการเปิดแชท—all ต้องเกิดขึ้นแบบ real-time และมี latency ต่ำ ระบบ Backend ต้องสร้างโครงสร้างแบบ Distributed และใช้เทคนิคอย่าง caching, prefetching, load balancing และ sharding เพื่อให้การตอบสนองทำได้รวดเร็วแม้มีผู้ใช้หลายล้านคนออนไลน์พร้อมกัน

QA-2 : Security

Scenario: ป้องกันการเข้าถึงข้อมูลผู้ใช้โดยไม่ได้รับอนุญาต

ส่วน	รายละเอียด
Source	ผู้ใช้ประสงค์ร้าย (Attacker) หรือบุคคลภายนอก
Stimulus	พยายามเข้าถึงข้อมูล เช่น โปรไฟล์ ข้อความ หรือระบบหลังบ้าน ผ่าน API หรือช่องโหว่
Artifact	Authentication Service, Database, Secure API Gateway
Environment	อินเทอร์เน็ตทั่วไป ที่มีความเสี่ยงด้านความปลอดภัย
Response	ระบบต้องปฏิเสธคำขอที่ไม่ได้รับสิทธิ์ พร้อมบันทึกเหตุการณ์เพื่อให้ทีม Security ตรวจสอบ
Response Measure	Authentication ต้องเป็นไปตามมาตรฐาน OAuth2/JWT

คำอธิบายเพิ่มเติม :

Tinder เป็นระบบที่มีข้อมูลส่วนตัวสำคัญ เช่น ชื่อ ที่ตั้ง อายุ รูปภาพ และบทสนทนา ดังนั้นความปลอดภัย (Security) ถือเป็นคุณสมบัติที่สำคัญมาก ระบบต้องป้องกันการเข้าถึงข้อมูลโดยไม่ได้รับอนุญาตและป้องกันภัยคุกคามรูปแบบต่างๆ เช่น brute-force, reverse engineering, API tampering, และการดักฟังข้อมูล

เพื่อให้มั่นใจว่าข้อมูลผู้ใช้ปลอดภัย Tinder ต้องใช้การเข้ารหัสข้อมูลทั้งระหว่างส่ง (TLS/HTTPS) และในระบบฐานข้อมูล ผู้ใช้ต้องผ่านกระบวนการยืนยันตัวตนที่แข็งแรง (เช่น OTP) นอกจากนี้ระบบยังต้องมีการตรวจจับกิจกรรมผิดปกติและมีระบบ Audit Log เพื่อให้ทีมงานตรวจสอบได้ทันทีหากเกิดเหตุการณ์ที่มีความเสี่ยง

QA-3 : Availability

Scenario: ป้องกันการเข้าถึงข้อมูลผู้ใช้โดยไม่ได้รับอนุญาต

ส่วน	**รายละเอียด**
Source	ผู้ใช้ทุกคนของระบบ
Stimulus	ผู้ใช้เปิดแอปในช่วงเวลาต่างๆ รวมถึงช่วง maintenance หรือช่วงที่เซิร์ฟเวอร์บางส่วนล่ม
Artifact	Load Balancer, Distributed Servers, Failover System
Environment	ระบบกระจายตัวบน Cloud (Distributed Cloud Environment)
Response	ระบบต้องยังคงให้บริการได้ต่อเนื่องโดยผู้ใช้ไม่รู้สึกถึงผลกระทบ
Response Measure	ระบบมี uptime ≥ 99.9%

คำอธิบายเพิ่มเติม :

ความพร้อมใช้งาน (Availability) เป็นสิ่งจำเป็นสำหรับแอปที่มีผู้ใช้งานทั่วโลกอย่าง Tinder ซึ่งผู้ใช้อาจอยู่ในโซนเวลาต่างกัน และแอปต้องพร้อมให้บริการ 24 ชั่วโมง ระบบจึงต้องถูกออกแบบให้มี uptime สูงและไม่ล่มแม้ระบบบางส่วนจะมีปัญหา

ในระดับสถาปัตยกรรม Tinder ใช้ระบบ Cloud แบบ Distributed รวมถึงมีการสำรองเซิร์ฟเวอร์ (replication) และระบบ failover อัตโนมัติ หากเซิร์ฟเวอร์หนึ่งล่ม ระบบจะเปลี่ยนเส้นทางไปยังเซิร์ฟเวอร์อื่นทันทีโดยไม่กระทบผู้ใช้ ระบบยังต้องรองรับ maintenance ระหว่างใช้งานโดยไม่กระทบการปัดหรือการแชทของผู้ใช้

QA-4 : Usability

Scenario: ผู้ใช้ใหม่ต้องสามารถเริ่มใช้งาน Tinder ได้ง่ายและเข้าใจการปัดทันที

ส่วน	**รายละเอียด**
Source	ผู้ใช้ใหม่ (New Users)
Stimulus	เปิดแอปครั้งแรก และลองปัดโปรไฟล์ครั้งแรก
Artifact	User Interface, Onboarding Flow, Swipe Interaction
Environment	ใช้งานบนมือถือทั้ง iOS/Android
Response	ผู้ใช้สามารถเข้าใจวิธีการปัดซ้าย/ขวา การเปิดโปรไฟล์ และการ Match ได้โดยไม่ต้องอ่านคู่มือ
Response Measure	ผู้ใช้ใหม่ 95% เข้าใจวิธีใช้งานภายใน 30 วินาที

คำอธิบายเพิ่มเติม :

Usability คือความง่ายในการใช้งาน ซึ่ง Tinder ให้ความสำคัญมาก เนื่องจากผู้ใช้จำนวนมากเป็นผู้ใช้หน้าใหม่ที่ไม่คุ้นเคยกับแอปเดตติ้งอื่นๆ การออกแบบต้องเรียบง่าย สื่อสารชัดเจน และให้ผู้ใช้เข้าใจได้ทันทีตั้งแต่ครั้งแรกที่เปิดใช้

อินเทอร์เฟซแบบปัดซ้าย (ไม่สนใจ) และปัดขวา (สนใจ) เป็น interaction ที่เข้าใจง่ายตามธรรมชาติ ทำให้ผู้ใช้สามารถเริ่มใช้ได้โดยไม่ต้องอ่านคู่มือ ระบบ onboarding ถูกลดจำนวนขั้นตอนให้เหลือเพียงไม่กี่คลิก ทำให้การเริ่มต้นใช้งานเป็นไปอย่างราบรื่น นอกจากนี้ UI ยังต้องเหมาะกับมือถือทุกรุ่น ขนาดหน้าจอ และรองรับทั้ง iOS/Android

QA-5 : Scalability

Scenario: ระบบรองรับการขยายตัวเมื่อมีผู้ใช้เพิ่มขึ้นอย่างรวดเร็ว เช่น หลังจากมีแคมเปญโฆษณาหรือไวรัล

ส่วน	**รายละเอียด**
Source	ผู้ใช้จำนวนมากที่สมัครและใช้งานพร้อมกัน
Stimulus	การเพิ่มขึ้นของ Request เช่น การโหลดโปรไฟล์ การส่งข้อความ การปัดต่อเนื่อง
Artifact	Microservices, Auto-scaling Servers, Database Cluster
Environment	ระบบ Cloud ที่รองรับ Horizontal Scaling
Response	ระบบสามารถขยาย Server อัตโนมัติและรองรับโหลดสูงได้โดยไม่ล่ม
Response Measure	รองรับการเพิ่มโหลดทันทีภายใน 10 วินาที

คำอธิบายเพิ่มเติม :

Tinder มีการเติบโตอย่างรวดเร็วจากการตลาดและไวรัล ดังนั้น Scalability หรือความสามารถในการรองรับผู้ใช้จำนวนมากจึงเป็นสิ่งที่จำเป็น ระบบต้องสามารถขยายขนาด (scale-out) ได้โดยอัตโนมัติเมื่อมีการใช้งานเพิ่มขึ้นแบบกะทันหัน เช่น หลังมีคอนเทนต์เกี่ยวกับ Tinder กลายเป็นไวรัล

สถาปัตยกรรมของ Tinder มักใช้ microservices ซึ่งช่วยให้แต่ละฟีเจอร์สามารถขยายได้อย่างอิสระ เช่น Matching Service อาจต้อง scale มากกว่า Chat Service ในบางเวลา ระบบต้องรองรับ auto-scaling และ CDN caching เพื่อรองรับผู้ใช้กระจายทั่วโลกโดยไม่ทำให้ระบบล่มหรือตอบสนองช้า

QA-6 : Modifiability

Scenario: การปรับแก้หรือเพิ่มฟีเจอร์ของระบบ Tinder ต้องทำได้อย่างรวดเร็ว โดยไม่กระทบส่วนอื่นของระบบ

ส่วน	**รายละเอียด**
Source	ทีมพัฒนา (Developers) และทีมสถาปัตยกรรมระบบ
Stimulus	ต้องการเพิ่มฟีเจอร์ใหม่ เช่น เพิ่มประเภทการปัดใหม่ (Super Boost แบบใหม่), ฟีเจอร์ความปลอดภัย, หรือปรับ Recommendation Algorithm
Artifact	Codebase, Microservices, API Gateway, Matching Engine
Environment	โครงสร้างสถาปัตยกรรมแบบ Microservices บน Cloud ที่มีบริการจำนวนมากรันอยู่
Response	ทีมพัฒนาสามารถแก้ไข เพิ่ม หรือลบฟีเจอร์ได้โดยไม่กระทบบริการอื่น และสามารถ deploy ส่วนที่แก้ได้แบบแยกอิสระ
Response Measure	การแก้ไขฟีเจอร์ใช้เวลาไม่เกิน 1–2 สปรินต์

คำอธิบายเพิ่มเติม :

- ผลกระทบของโค้ดเก่าไม่เกิน 5% ของ service ที่ใช้งานร่วมกัน
- สามารถ deploy service เดียวได้โดยไม่ต้อง deploy ทั้งระบบ (Independent Deployment)
- Downtime จากการปรับปรุงระบบ = 0 วินาที (ใช้ Rolling Deployment / Blue-Green Deployment) |

ส่วนที่ 4 : Constraints

Technical Constraints

TC-01 : รองรับผู้ใช้งานพร้อมกันจำนวนมาก (High Concurrency Support)

ระบบต้องรองรับผู้ใช้หลายล้านคนออนไลน์พร้อมกัน จึงต้องใช้สถาปัตยกรรมแบบ Distributed Microservices และระบบ Cloud ที่สามารถ scale-out ได้อย่างรวดเร็ว

TC-02 : Latency ต้องต่ำมาก (Low-latency Requirement)

ฟีเจอร์หลัก เช่น Swipe, Match, และ Messaging ต้องตอบสนองภายใน < 1 วินาที ทำให้ต้องใช้

- Caching Layer
- Asynchronous Processing
- Load Balancing เพื่อให้การตอบสนองรวดเร็วในทุกภูมิภาค

TC-03 : Mobile-first Architecture

แอป Tinder ต้องทำงานได้ดีบน iOS และ Android ทำให้การออกแบบ API, UI และ Interaction ต้องรองรับการใช้งานบนมือถือเป็นหลัก

TC-04 : Data Consistency Across Services

เนื่องจาก Tinder ใช้ระบบกระจายข้อมูลหลายโหนด การเก็บข้อมูลผู้ใช้และการจับคู่ต้องใช้เทคนิค eventual consistency และระบบ messaging เช่น Kafka

Time Constraints

TI-C-01 : การออกฟีเจอร์ใหม่ต้องทำได้เร็ว (Rapid Feature Deployment)

ตลาด dating applications มีการแข่งขันสูง จึงต้องสามารถเพิ่ม/แก้ไขฟีเจอร์ได้ภายใน รอบสปรินต์ 1–2 สัปดาห์ โดยไม่ทำให้ระบบล่ม

ส่งผลให้ต้องใช้

- CI/CD Pipeline
- Feature Flag
- A/B Testing

TI-C-02 : ระบบต้องพร้อมใช้งาน 24/7 (No Downtime Requirements)

ผู้ใช้กระจายทั่วโลก ทำให้ Tinder ไม่สามารถหยุดบริการเพื่อ Maintenance ได้ ต้องใช้

- Blue-Green Deployment
- Rolling Updates

Budget Constraints

BC-01 : ค่าใช้จ่ายโครงสร้างพื้นฐานสูง (High Infrastructure Costs)

เนื่องจากต้องรองรับ Global Scale จึงต้องใช้

- Cloud Providers หลายภูมิภาค
- Auto-scaling Servers
- High-performance Databases ทำให้มีข้อจำกัดเรื่องการควบคุมงบประมาณด้าน Server, Storage, และ Network

BC-02 : ค่าใช้จ่ายด้าน AI/ML Model Training

Recommendation Engine และ Matching Algorithm ต้องใช้ทรัพยากรประมวลผลสูง เช่น GPU clusters ซึ่งมีต้นทุนสูงต่อครั้ง

Legal/Policy Constraints

LC-01 : ข้อกำหนดด้านความเป็นส่วนตัว (Privacy Regulations)

Tinder ต้องปฏิบัติตามกฎหมายความเป็นส่วนตัวของหลายประเทศ เช่น

- GDPR (Europe)
- CCPA (California)
- PDPA (Thailand)

LC-02 : ข้อจำกัดด้านอายุผู้ใช้ (Age Restrictions)

แอปห้ามให้บริการแก่ผู้ใช้ที่อายุต่ำกว่า 18 ปีตามกฎหมายหลายประเทศ
ทำให้ต้องมีระบบยืนยันอายุและคัดกรองบัญชีที่ไม่ผ่านเกณฑ์

LC-03 : ข้อกำหนดด้าน Content Moderation

Tinder ต้องมีระบบควบคุมเนื้อหาไม่เหมาะสมตามนโยบายแพลตฟอร์มและกฎหมาย เช่น

- การตรวจจับภาพลามก
- การล่วงละเมิด
- Hate Speech ซึ่งจำเป็นต้องมี AI และทีม Moderation

ส่วนที่ 5 Assumptions

Assumption 1 : ผู้ใช้มีการเชื่อมต่ออินเทอร์เน็ตตลอดเวลา

ระบบ Tinder ถูกออกแบบให้ทำงานแบบ real-time
ดังนั้นสมมติว่าผู้ใช้ต้องมีการเชื่อมต่ออินเทอร์เน็ตที่เสถียรเพื่อให้สามารถโหลดโปรไฟล์ ปัด และส่งข้อความได้อย่างต่อเนื่อง

Assumption 2 : อุปกรณ์ของผู้ใช้รองรับการใช้งานแอป Tinder

สมมติว่าอุปกรณ์ของผู้ใช้ เช่น สมาร์ตโฟน iOS/Android มีสเปกขั้นต่ำและระบบปฏิบัติการเวอร์ชันที่รองรับ โดยสามารถรันฟีเจอร์กราฟิก การปัด การแสดงผลภาพ และการแจ้งเตือนแบบ Push Notification ได้

Assumption 3 : ผู้ใช้กรอกข้อมูลโปรไฟล์จริงตามความเป็นจริงระดับหนึ่ง

ระบบจับคู่ของ Tinder อาศัยข้อมูลจากโปรไฟล์ เช่น อายุ ความสนใจ และตำแหน่งเพื่อให้แนะนำคู่ที่เหมาะสม จึงสมมติว่าผู้ใช้กรอกข้อมูลที่ไม่บิดเบือนจนทำให้ระบบทำงานผิดพลาด (แม้จะมีระบบตรวจสอบเพิ่มเติมก็ตาม)

Assumption 4 : ระบบภายนอก (External Services) มีความพร้อมให้บริการ Tinder ใช้ระบบภายนอก เช่น

- **Firebase / Push Notification Service**
- **ระบบยืนยันตัวตนผ่าน SMS**
- **การเชื่อมต่อกับ Instagram / Spotify**

จึงสมมติว่าบริการเหล่านี้มีความพร้อมและออนไลน์ตลอดเวลาไม่เกิด downtime ที่ยาวนาน

Assumption 5 : ปริมาณข้อมูลและจำนวนผู้ใช้มีการเติบโตต่อเนื่อง

สมมติว่าจำนวนผู้ใช้เพิ่มขึ้นอย่างสม่ำเสมอ จึงต้องออกแบบระบบให้รองรับการขยายตัว (Scalable) เพื่อรองรับโหลดจำนวนมากในอนาคต โดยไม่ต้องออกแบบใหม่ตั้งแต่ต้น

ส่วนที่ 6 Priority & Trade-offs

Quality Attributes Priority

Rank	Quality Attribute	เหตุผล
1	Performance	Tinder ต้องโหลดโปรไฟล์และตอบสนองการปัดอย่างรวดเร็ว ถ้าช้าเกิน 1–2 วินาที ผู้ใช้จะออกจากแอปทันที ส่งผลโดยตรงต่อประสบการณ์และรายได้
2	Availability	แอปต้องออนไลน์ 24/7 ทั่วโลก หากล่มแม้ไม่กี่นาที ผู้ใช้จะไม่สามารถปัดหรือแชทได้ ทำให้สูญเสียโอกาสทางธุรกิจและความเชื่อมั่น
3	Scalability	ผู้ใช้ Tinder เพิ่มขึ้นอย่างรวดเร็ว และมี peak load สูงมาก โดยเฉพาะช่วงเย็น-ค่ำ การ scale-out เป็นหัวใจสำคัญเพื่อรองรับ user demand
4	Security	แอปจัดการข้อมูลส่วนตัวและข้อมูลแชท ต้องป้องกันการรั่วไหลและการเข้าถึงโดยไม่ได้รับอนุญาต เป็นข้อกำหนดสำคัญตามกฎหมายหลายประเทศ
5	Usability	การใช้งานของ Tinder ต้อง simple, เข้าใจง่าย และ onboarding ไม่ซับซ้อน เพื่อดึงดูดผู้ใช้ใหม่และคงผู้ใช้เดิม
6	Modifiability	Tinder เพิ่มฟีเจอร์ใหม่บ่อย (เช่น Super Boost, Explore Mode) ต้องแก้ไขหรือเพิ่มฟีเจอร์ได้เร็ว แต่ความสำคัญรองจาก Performance/Availability

Trade-offs Analysis

Trade-off #1: Performance vs Security

อธิบาย :

- Performance ต้องการการตอบสนองเร็ว เช่น โหลดโปรไฟล์ทันทีหลังปัด
- Security ต้องตรวจสอบสิทธิ์, เข้ารหัสข้อมูล, ตรวจสอบพฤติกรรมผิดปกติ ซึ่งทำให้ latency เพิ่มขึ้น

ตัวอย่าง :

- การเข้ารหัสข้อมูล (Encryption) ทำให้ประมวลผลช้าลง
- การเช็ค Token ทุก request ทำให้ response เพิ่มขึ้นเล็กน้อย
- การตรวจสอบรูปภาพไม่เหมาะสม (Image Moderation) ใช้เวลาประมวลผลมาก

Decision: เลือก Performance เป็นอันดับแรก แต่ไม่ลดทอน Security ที่จำเป็น

เพราะ Performance คือประสบการณ์หลักของผู้ใช้ Tinder ส่วน Security ต้องมีในระดับป้องกันความเสี่ยง แต่จะใช้เทคนิคช่วย เช่น

- Caching Tokens
- ใช้ CDN
- ใช้ asynchronous moderation เพื่อให้ security ไม่กระทบ performance โดยตรง

Trade-off #2: Scalability vs Cost

อธิบาย

- Scalability ต้องการเซิร์ฟเวอร์จำนวนมาก กระจายหลาย region รองรับ peak load หลักล้าน
- Cost จะเพิ่มสูงมากตามจำนวนเครื่อง, bandwidth, database replication

ตัวอย่าง

- Auto-scaling บางครั้ง scale มากเกินช่วง peak ⟶ ค่าใช้จ่ายสูง
- Database sharding และ read replicas ทำให้ค่า storage สูงขึ้น

Decision: เลือก Scalability เพื่อรองรับการเติบโตของผู้ใช้

เพราะถ้าระบบช้าหรือใช้งานไม่ได้จะกระทบผู้ใช้ทันที และทำให้เสียรายได้

แต่มีมาตรการควบคุมต้นทุน เช่น

- ใช้ spot instances (ที่ถูกกว่า)
- ใช้ autoscaling แบบ predictive
- ลด cost ผ่าน caching (Redis, CDN)

Trade-off #3: Modifiability vs System Complexity

อธิบาย

- Modifiability ต้องการ microservices แยกเป็นบริการย่อย เพื่อแก้/เพิ่มฟีเจอร์โดยไม่กระทบระบบ

แต่ microservices จำนวนมากทำให้

- ระบบซับซ้อนขึ้น
- ต้องจัดการการสื่อสารระหว่าง service
- ต้องมี DevOps และ monitoring มากขึ้น

ถ้าระบบใหญ่เกินไป การ maintain ก็ยากขึ้น

Decision: ใช้ Microservices แบบแยกเฉพาะ Domain สำคัญ

ไม่แยกทุก service เกินความจำเป็น เช่น

- Matching, Messaging แยก
- Recommendation แยก
- แต่บาง service รวมกัน เช่น user settings + privacy management

ผลลัพธ์: Modifiable พอเหมาะ และ complexity ไม่เกินควบคุม

Trade-off #4: Availability vs Deployment Speed

อธิบาย

- ต้องการ Availability สูง (99.9%+) ⟶ ห้าม downtime
- แต่ต้องการ Deploy ฟีเจอร์บ่อย ⟶ มีโอกาสเกิด bug และความเสี่ยงระบบล่ม

Decision: ใช้ Blue-Green Deployment และ Feature Flags

ทำให้สามารถ deploy ได้โดยไม่หยุดระบบ และย้อนกลับได้ทันทีหากเกิดปัญหา

Trade-off #5: Usability vs Functionality

อธิบาย

- **Usability เน้นใช้งานง่าย ปัดซ้ายขวาไม่ซับซ้อน**
- **Functionality เพิ่มฟีเจอร์เยอะ เช่น Explore Mode, Photo Verification, Boost**
  แต่ฟีเจอร์ที่มากขึ้นอาจทำให้ UI ซับซ้อนขึ้น

Decision: Usability มาก่อน แต่เพิ่มฟีเจอร์แบบค่อยเป็นค่อยไป

โดยซ่อนฟีเจอร์รองไว้ในเมนูย่อยแทน เพื่อไม่ให้รบกวน core flow ของการปัด