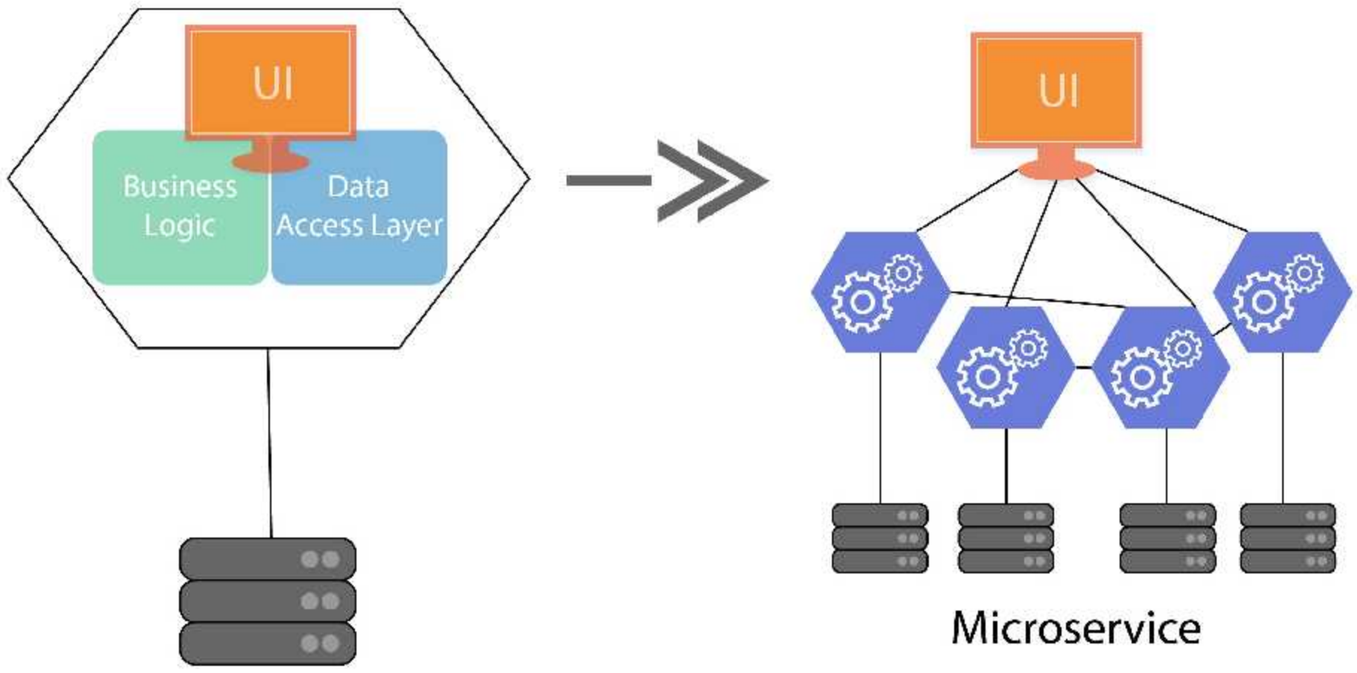
ANALISIS DESAIN

BERBASIS ARSITEKTUR MICROSERVICE DENGAN REPRESENTATIONAL STATE TRANSFER (REST)

(STUDI KASUS : HEARTENLY.COM)

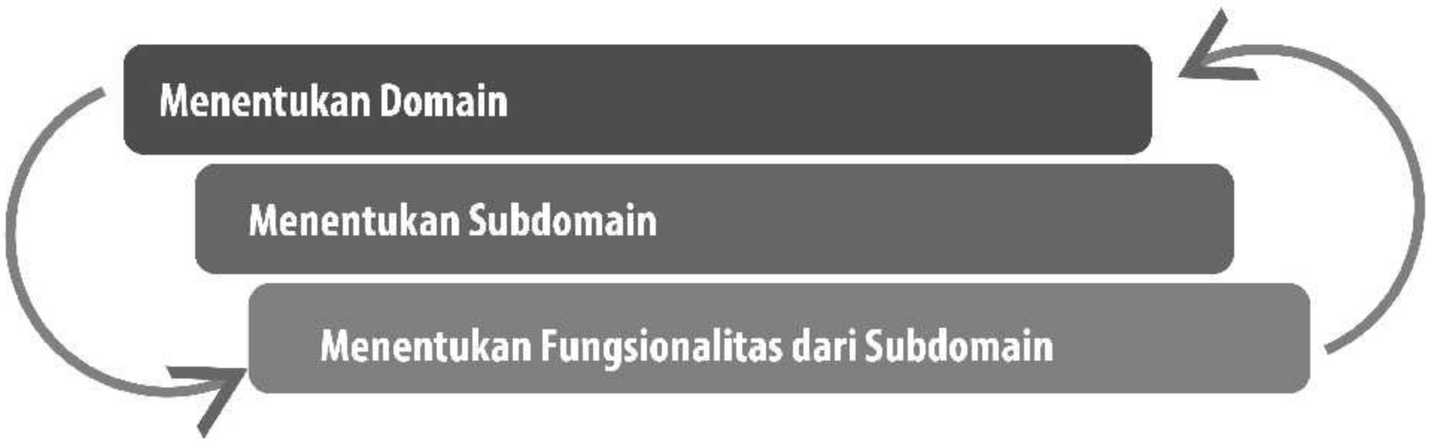
KAUTSAR TISAMAWI - 130117327



Monolitik

Microservice

Untuk beralih dari arsitektur Monolitik ke Microservice memerlukan metode dekomposisi. Dekomposisi digunakan untuk memecah layanan yang tadinya besar menjadi kecil. Metode tersebut menggunakan prinsip Domain-driven Design, berikut adalah tahapan-tahapannya:



Heartenly
heartenly.com

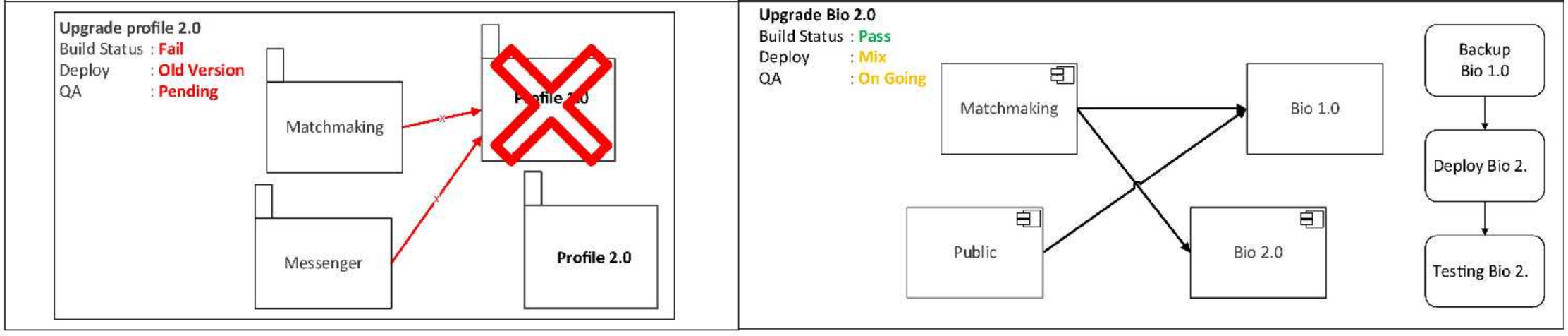
Heartenly.com merupakan salah satu aplikasi pencari jodoh asal Indonesia, yang saat ini mempunyai jumlah pengguna lebih dari 300.000 dan masih terus bertambah. Heartenly.com berencana untuk menambahkah layanan-layanan baru dan memperbaiki yang sudah ada agar lebih memanjakan penggunanya. Namun saat ini Heartenly.com masih menggunakan arsitektur monolitik. Monolitik harus mematikan seluruh layanan ketika akan menambah atau memperbaiki suatu layanan tertentu dan tidak dapat melakukan upgrade secara independen. Untuk itu, sebelum melakukan penambahan atau peningkatan layanan, Heartenly.com harus mempersiapkan arsitekturnya terlebih dahulu agar mudah untuk melakukan peningkatan atau perbaikan layanan. Arsitektur tersebut adalah arsitektur Microservice

Rumusan Masalah

1. **Bagaimana mendesain arsitektur Microservices dengan REST pada Heartenly.com?**
2. **Bagaimana perbedaan kemudahan upgrade antara arsitektur Monolitik dengan Microservice?**

Tujuan

1. **Mendesain arsitektur Microservice dengan REST pada Heartenly.com**
2. **Membandingkan kemudahan upgrade arsitektur Monolitik dan Microservice**



Pada gambar diatas pada bagian monolitik terdapat package Matchmaking, Messenger, Profile 1.0 dan Profile 2.0. Gambar tersebut menggambarkan, ketika Matchmaking dan Messenger akan menggunakan Profile 2.0 yang merupakan versi terbaru dari Profile 1.0, aplikasi tidak dapat di-deploy, seluruh sistem harus dimatikan, dan dipastikan terlebih dahulu ter-compile untuk kemudian maju ke tahap QA, deploy dan testing. Berbeda dengan microservice tahap QA, deploy dan testing dapat dilakukan secara independen, pada microservice tertentu tanpa mempengaruhi sistem yang lain.