

PENERAPAN *CONTINUOUS INTEGRATION*

Studi Kasus : MedRecApp

Laporan Tugas Akhir

Disusun sebagai syarat kelulusan

Program Alih Jenjang D3-D4 Teknologi Informasi Kesehatan

Oleh

Fachrul Pralienka Bani Muhammad

Hernawati Susanti Samosir

Yuanita Annisatul Hijriyah



**PROGRAM ALIH JENJANG D3-D4
TEKNOLOGI INFORMASI KESEHATAN
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2014**

LEMBAR PENGESAHAN

ABSTRAK

KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas rahmat-Nya yang menyertai penulis selama pengerjaan Tugas Akhir ini, sehingga proses pengerjaan Tugas Akhir ini sesuai dengan perencanaan dan penyusunan laporan Tugas Akhir ini dapat dilalui dengan baik. Laporan Tugas Akhir ini ditulis sebagai bagian dari syarat kelulusan Diploma 4 STEI ITB.

Penulisan laporan Tugas Akhir ini bertujuan untuk memberi pengetahuan kepada pembaca tentang metode pembangunan perangkat lunak *Continuous Integration* dengan studi kasus aplikasi rumah sakit MedRecApp.

Penulis mengucapkan terimakasih kepada pihak-pihak yang telah membantu penulis dalam penyelesaian tugas akhir ini, di antaranya:

1. Ibu Dr. Inggriani Liem, selaku pembimbing I yang banyak memberikan arahan dan bimbingan sehingga tugas akhir ini dapat selesai.
2. Bapak Yudistira Dwi Wardhana Asnar, S.T., PhD. Selaku pembimbing II yang memberikan arahan dan bimbingan sehingga tugas akhir ini dapat selesai.
3. Koordinator D4 STEI ITB, Dr. Ir. G.A. Putri Saptawati, M. Comm.
4. Bagian Administrasi D4 STEI ITB.
5. ITD (Institut Teknologi Del) sebagai sumber informasi yang membantu penyelesaian tugas akhir ini.
6. Orangtua yang selalu memberikan motivasi dalam pengerjaan tugas akhir ini.
7. Teman-teman seperjuangan dan teman-teman lainnya yang tidak dapat penulis sebutkan satu persatu yang membantu penyelesaian tugas akhir ini.

Seperti peribahasa “tidak ada gading yang tidak retak”. Penulis pun menyadari bahwa di dalam laporan Tugas Akhir masih terdapat kesalahan, oleh karena itu penulis sangat mengharapkan kritik dan saran yang bersifat membangun untuk perbaikan dokumen ini.

BAB I

PENDAHULUAN

1.1. Latar Belakang

Rekayasa perangkat lunak adalah suatu strategi pengembangan perangkat lunak yang melingkupi lapisan proses, metode, dan *tools***Error! Reference source not found..** Siklus hidup pengembangan perangkat lunak terdiri tahapan *requirement, design, code, testing, deployment* dan *maintenance***Error! Reference source not found..** Pada pembangunan perangkat lunak skala besar, setiap penambahan fitur akan dilakukan secara *increment*. Penambahan fitur tersebut akan melewati siklus hidup pengembangan perangkat lunak beberapa kali hingga produk versi tertentu dirilis. Berdasarkan survei pengembangan perangkat lunak perusahaan mengeluarkan biaya paling besar untuk *maintenance* **Error! Reference source not found..**

Fakta yang terjadi pada saat membangun perangkat lunak adalah kebutuhan *customer* terhadap perangkat lunak yang sering berubah dan *customer* tidak ingin menunggu lebih lama lagi untuk memperoleh perangkat lunak.

Pada dasarnya sebuah perangkat lunak dibangun dengan memperhatikan dua fokus utama [49]. Di antaranya: (1)Seberapa besar biaya yang dibutuhkan untuk membangun perangkat lunak? (2)Berapa lama waktu yang dibutuhkan untuk membangun perangkat lunak? Kedua hal tersebut memiliki keterikatan satu sama lain. **Permasalahan yang sering terjadi pada pembangunan perangkat lunak adalah produk *error* pada tahap integrasi.** Jika proses penanganan *error* tersebut adalah manual maka dibutuhkan *effort* maksimal dan waktu jangka panjang untuk memperbaiki *product* tersebut.

Otomatisasi adalah kunci untuk melakukan proses yang sama berulang kali. **Dengan penerapan otomatisasi maka proses *build, deploy, testing* dan rilis dapat dilakukan dengan cepat.** Pengintegrasian perangkat lunak juga harus dilakukan berulang kali agar dapat mengurangi resiko dan memperbaiki kualitas perangkat lunak**Error! Reference source not found..** Pengintegrasian tersebut dapat dilakukan dengan menggunakan praktik *Continuous Integration* (CI).

CI adalah praktik pengembangan perangkat lunak yang mengintegrasikan hasil pekerjaan anggota tim secara rutin untuk menemukan kesalahan pada proses integrasi secepat mungkin. **Error! Reference source not found.** Dengan penerapan *continuous integration* maka perangkat lunak *customer* dipastikan **dapat** bekerja terhadap setiap perubahan baru. Jika ada kesalahan pada proses integrasi, maka tim dapat memperbaikinya dengan cepat. Tim yang menggunakan *continuous integration* secara efektif mampu mendeteksi *bug* lebih awal, menghasilkan perangkat lunak lebih cepat dengan sedikit *bug*, mengurangi biaya dan jangka waktu perbaikan perangkat lunak pada proses *delivery* dibandingkan tim yang tidak menggunakan *continuous integration*. **Error! Reference source not found.**

Continuous Integration akan diterapkan dan dilakukan pengamatan pada aplikasi MedRecApp yang dibangun oleh sebuah tim. Tidak adanya *automated testing* pada tahap integrasi mengakibatkan sulitnya memperoleh produk versi tertentu tanpa cacat, meskipun tim telah menggunakan *version control*. Dari segi kegunaan, aplikasi tersebut memiliki potensi besar untuk dapat dikembangkan dengan fitur-fitur baru yang sesuai dengan kebutuhan rumah sakit X.

Berdasarkan uraian tersebut, maka judul tugas akhir yang diangkat adalah "Penerapan *Continuous Integration* pada aplikasi MedrecApp".

1.2. Tujuan

Berdasarkan latar belakang yang telah diuraikan sebelumnya, Tugas Akhir ini bertujuan untuk membentuk kerangka kerja yang meliputi prosedur, teknik, *toolset* yang mendukung *continuous integration* aplikasi MedrecApp berbasis Java *desktop*.

1.3. Rumusan Masalah

Pada tugas akhir ini, rumusan masalah yang diangkat adalah bagaimana membentuk kerangka kerja yang meliputi prosedur, teknik, dan *toolset* yang mendukung *continuous integration* pada aplikasi MedrecApp berbasis Java *desktop*?

1.4. Ruang Lingkup Masalah

Ada beberapa ruang lingkup pada *Continuous integration* yang dilakukan, diantaranya:

1. Mencakup *Java desktop application* yang sudah ada yang terdiri dari modul master, modul *front office*, modul poliklinik dan modul rekam medis. Proses *development* mencakup *compile source program*, membuat dan menjalankan *unit test* dan *integration test*
2. Prosedur yang digunakan pada CI mencakup:
 - a. Penentuan komitmen kesepatan awal
 - b. Penggunaan VCS
 - c. Penggunaan *automated build*
3. Teknik yang digunakan untuk *version control system*, *automated testing*, *continuous integration* adalah:
 - a. VCS : *distributed version control system* dengan alur kerja *Centralized workflow*
 - b. *Unit test* dan *Integration testing*: *Automated testing* dan *Bottom-up integration*
 - c. *Build tool* : *scripting* yang dihasilkan dari Netbeans dan dimodifikasi
 - d. *Continuous Integration* : menggabungkan teknik dari VCS, *testing* dan *build*
4. *Tools* yang digunakan untuk *version control system*, *automated testing*, *continuous integration* adalah *open source* dan *free software* dengan rincian sebagai berikut:
 - a. *Version Control* dan *Software hosting* : Git dan Github
 - b. *Unit Testing* dan *Integration testing*: Junit dan Fest
 - c. *Automated build* dan *CI* : Ant dan Jenkins
5. Tidak mencakup *inspection code* dan *deployment* pada *Continuous Integration*

1.5. Metodologi

Pada bagian ini diuraikan mengenai metodologi penelitian sebagai berikut:

1. Studi Literatur

Hal yang perlu dilakukan untuk melakukan studi literatur sebagai berikut:

- a. Mengumpulkan, mempelajari dan memahami bahan dan konsep *continuous integration*
- b. Memilih *sample* aplikasi MedRecApp berbasis Java Desktop yang telah dikembangkan oleh sebuah tim sebagai studi kasus penerapan *continuous integration*

- c. Mempelajari, memahami dan menentukan prosedur, teknik dan *toolset* yang mendukung *continuous integration* pada aplikasi MedRecApp
- 2. Analisis
 - Menentukan ide pemikiran kerangka kerja yang mendukung *continuous integration* pada aplikasi MedRecApp
- 3. Implementasi dan Pengujian
 - a. Menerapkan prosedur dan teknik dengan menggunakan *toolset* yang ada pada VCS, *testing*, *build* yang mendukung *continuous integration*
 - b. Menguji alur *continuous integration* pada aplikasi MedRecApp
 - c. Kendala yang dihadapi pada saat melakukan *continuous integration*

4. Penutup

Berisi kesimpulan yang dapat diperoleh dari hasil implementasi dan pengujian CI pada aplikasi MedrecApp berbasis Java *desktop* dan saran untuk perbaikan penelitian selanjutnya.

