BAB IV
STUDI KASUS *CONTINUOUS INTEGRATION* SECARA MANUAL DAN MENGGUNAKAN *TOOLSET*

Pada bab ini akan dijelaskan tentang penerapan konsep pembangunan perangkat lunak dengan *continuous integration* yang dilakukan secara manual dan menggunakan *toolset*. Penerapan konsep tersebut dilakukan untuk menunjukkan perbedaan proses dari keduanya. Praktik yang mencakup penerapan *continuous integration* secara manual yaitu praktik penyimpanan versi secara manual, praktik pengujian kode program secara manual, praktik eksekusi *build* secara manual, dan praktik pengintegrasian modul secara manual. Sedangkan praktik yang mencakup penerapan *continuous integration* dengan menggunakan *toolset* yaitu praktik penyimpanan versi dengan *version control system tool*, praktik pengujian kode program dengan *automated testing tool*, praktik eksekusi *build* dengan *automated build tool* dan praktik pengintegrasian modul dengan *automated continuous integration tool*.

Studi kasus yang digunakan untuk menerapkan konsep pembangunan perangkat lunak dengan *continuous integration* secara manual dan menggunakan *toolset* adalah aplikasi rekam medis berbasis *java desktop*, yang bernama medrecapp. Pembangunan aplikasi tersebut dilakukan oleh satu tim yang terdiri dari tiga orang *developer*, yaitu Fachrul, Hernawati, dan Yuanita.

4.1. *Package* aplikasi medrecapp

Pada sub bab ini akan dijelaskan tentang pembagian *package* aplikasi medrecapp untuk mendukung penerapan konsep *continuous integration*. Pembagian *package* tersebut dilakukan tim untuk meningkatkan produktivitas pekerjaan setiap *developer* (lihat **tabel 4-1**). Setiap *package* memiliki perbedaan area fungsional dengan *package* yang lain. Para *developer* akan lebih mudah membuat kode program berdasarkan pada satu area fungsional.

Tabel 4-1. Pembagian *package* aplikasi medrecapp

No	Daftar *packages*	*Developers*		
		Fachrul	Hernawati	Yuanita
1	Spesialis	✓		
2	Jaminan	✓		
3	Pasien		✓	
4	Staf			✓
5	Perawat	✓		
6	Dokter			✓
7	Tindakan		✓	
8	Rekam medis			✓
9	Pelayanan tindakan		✓	

Pada setiap *package* aplikasi medrecapp, terdapat kelas DAO (*Data Access Object*), kelas *entity*, kelas *GUI*, kelas *interface*, kelas *services*, dan kelas tabel model. Setiap kelas pada suatu *package* memiliki beberapa *method*. *Package* aplikasi medrecapp terdiri dari sembilan *package* yaitu *package* spesialis, *package* jaminan, *package* pasien, *package* staf, *package* perawat, *package* dokter, *package* tindakan, *package* rekam medis, dan *package* pelayanan tindakan. Dependensi antar *package* aplikasi medrecapp dapat dilihat pada **gambar 4-1**.

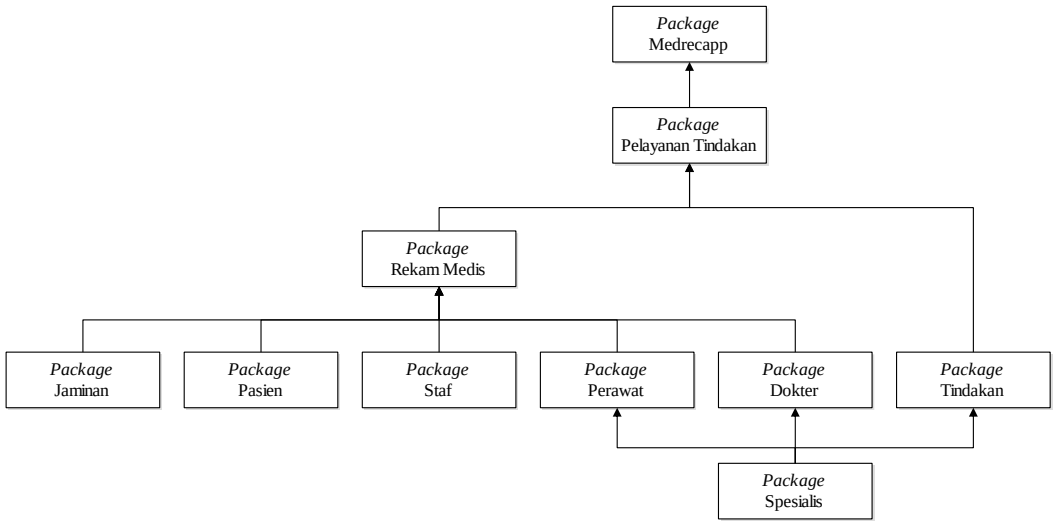
Dengan membagi pekerjaan berdasarkan *package*, para *developer* dapat dimudahkan dalam membuat kode program di satu area fungsional.

Untuk meningkatkan produktivitas pekerjaan setiap developer

Developer produktivitasnya berbeda, lebih mudah 1 modul area fungsional

Pada proses pembangunan aplikasi medrecapp dengan continuous integration, tim membagi pekerjaan berdasarkan package aplikasi medrecapp. Mereka membagi pekerjaan berdasdarkan package tersebut.

Pada sub bab ini akan dijelaskan pembagian pekerjaan yang dilakukan tim terhadap modul aplikasi medrecapp. Setiap developer akan membagi pekerjaan berdasarkan package pada aplikasi medrecapp.



Gambar 4-1. *Package* pada aplikasi medrecapp

4.2. Praktik pembangunan aplikasi medrecapp secara manual

Pada sub bab ini akan dijelaskan tentang praktik pembangunan aplikasi medrecapp dengan *continuous integration* tanpa menggunakan bantuan *toolset*.

4.2.1. Praktik penyimpanan versi secara manual

4.2.2. Praktik pengujian kode program secara manual

4.2.3. Praktik eksekusi *build* secara manual

4.2.4. Praktik pengintegrasian modul secara manual

4.3. Praktik pembangunan aplikasi medrecapp menggunakan *toolset*

Pada sub bab ini akan dijelaskan tentang praktik pembangunan aplikasi medrecapp dengan continuous integration dengan menggunakan bantuan toolset.

4.3.1. Praktik penyimpanan versi dengan *version control system tool*

4.3.2. Praktik pengujian kode program dengan *automated testing tool*

4.3.3. Praktik eksekusi *build* dengan *automated build tool*

4.3.4. Praktik pengintegrasian modul dengan *automated continuous integration tool*