Pengenalan Rekayasa Perangkat Lunak

Team Teaching Mata Kuliah RPL

Jurusan Teknologi Informasi

Politeknik Negeri Malang

Outline

- Pengembangan perangkat lunak
- Etika dalam rekayasa perangkat lunak

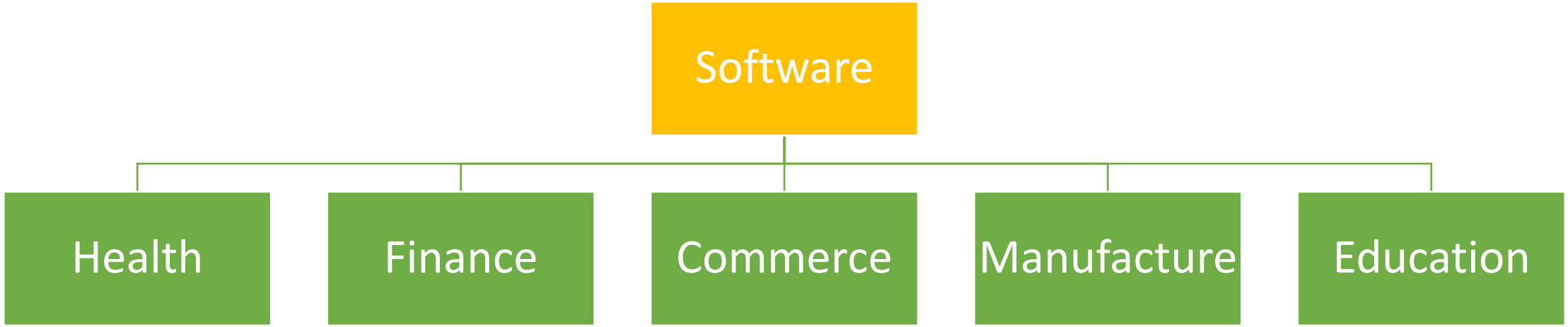
Tujuan

- Memahami konsep rekayasa perangkat lunak secara umum
- Memahami urgensitas penerapan rekayasa perangkat lunak
- Memahami aspek- aspek penting dalam rekayasa perangkat lunak

Perangkat Lunak (*software*)

- **Kakas** berupa **program** beserta **dokumentasi** terkait program
- Perangkat lunak (software) adalah serangkaian instruksi, data, atau program yang digunakan untuk mengoperasikan komputer dan menjalankan tugas-tugas tertentu.
- Software dikembangkan untuk pengguna khusus atau dapat untuk pasar/pengguna umum.

Cakupan software



Contoh perangkat lunak

Praktikum Basis Data (MSK)

Aktifitas | Setting | Reports | Question Bank | Peserta

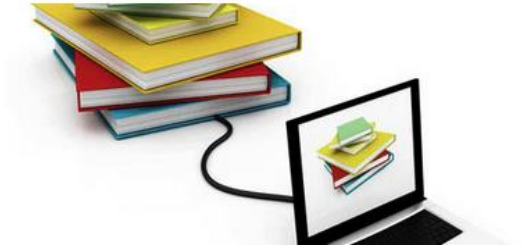
INFORMASI COURSE LMS | GOOGLE MEETING | ZOOM MEETING

Deskripsi MK	: Mata kuliah praktikum basis data TI-1H
Program Studi	: Diploma IV Teknik Informatika
Kelas Siakad	: 1H-TI
Jumlah Mahasiswa	: 30
URL LMS	: https://lmsslc.polinema.ac.id/course/view.php?id=6868

Manajemen Group

No	Nama Group	Peserta	Hapus
1.	Diploma IV Teknik Informatika 1H-TI	30	Hapus Group & Un-enroll pesertanya ✕

Big Data (MSK)

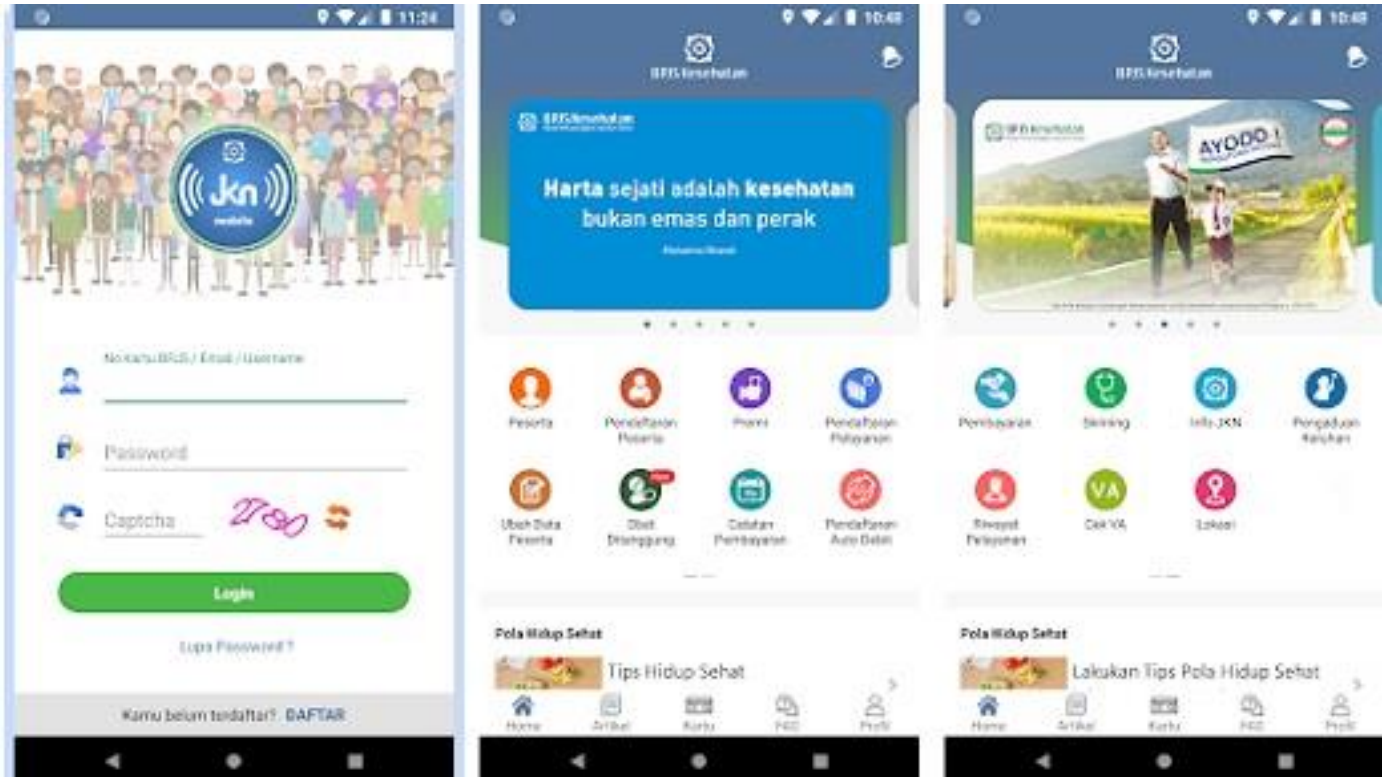


Aktifitas | Setting | Reports | Question Bank | Peserta

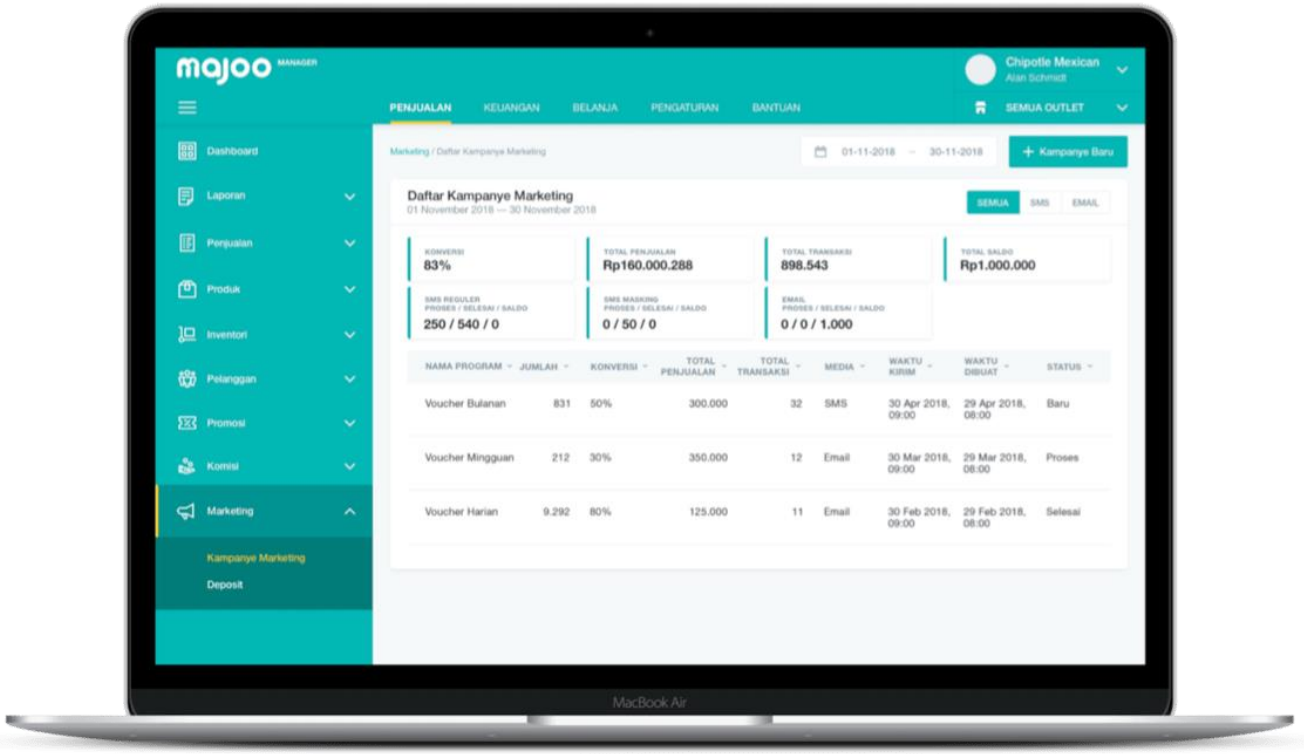
INFORMASI COURSE LMS | GOOGLE MEETING | ZOOM MEETING

Deskripsi MK	: Mata kuliah Big Data kelas TI-4J
Program Studi	: Diploma IV Teknik Informatika
Kelas Siakad	: 4J-TI

Contoh perangkat lunak



Contoh perangkat lunak



Contoh perangkat lunak

Jenis Perangkat Lunak

1. Perangkat Lunak Sistem (System Software)

Perangkat lunak sistem bertanggung jawab untuk mengelola perangkat keras komputer dan menyediakan platform untuk menjalankan perangkat lunak aplikasi.

- Sistem Operasi (Operating System)
- Driver Perangkat (Device Drivers)
- Utilitas (Utilities): Program yang melakukan tugas pemeliharaan dan manajemen sistem, seperti antivirus, defragmentasi disk, dan manajemen file.

Jenis Perangkat Lunak

2. Perangkat Lunak Aplikasi (Application Software)

Perangkat lunak aplikasi dirancang untuk membantu pengguna melakukan tugas tertentu atau memecahkan masalah.

- Aplikasi Produktivitas: Program yang membantu dalam pekerjaan sehari-hari
- Perangkat Lunak Kreatif: Program yang digunakan untuk keperluan desain grafis, pengeditan video, atau produksi musik.
- Aplikasi Web: Program yang berjalan di browser dan digunakan untuk berbagai keperluan.
- Perangkat Lunak Hiburan: Program untuk tujuan hiburan.

Jenis Perangkat Lunak

3. Perangkat Lunak Pengembangan (Development Software)

Perangkat lunak ini digunakan oleh pengembang untuk membuat perangkat lunak lain.

- Editor Kode: Alat untuk menulis dan mengedit kode sumber.
- Kompiler: Program yang mengubah kode sumber menjadi kode mesin yang dapat dijalankan oleh komputer.
- Integrated Development Environment (IDE): Lingkungan pengembangan terpadu yang menyediakan fasilitas untuk menulis, menguji, dan debugging perangkat lunak.
- Sistem Kontrol Versi: Alat untuk melacak perubahan kode sumber dan mengelola berbagai versi proyek perangkat lunak.

Jenis Perangkat Lunak

4. Perangkat Lunak Berbasis Jaringan (Network Software)

Jenis perangkat lunak ini memungkinkan komputer untuk berkomunikasi dan berbagi sumber daya melalui jaringan.

- Perangkat Lunak Server: Program yang menyediakan layanan kepada komputer lain di jaringan, seperti server web (Apache HTTP Server), server email (Microsoft Exchange Server).
- Perangkat Lunak Keamanan Jaringan: Alat yang melindungi jaringan dari ancaman seperti firewall, perangkat lunak VPN, dan program deteksi intrusi.

Jenis Perangkat Lunak

5. Perangkat Lunak Terbenam (Embedded Software)

Perangkat lunak terbenam diinstal dalam perangkat keras khusus dan digunakan untuk mengendalikan fungsi perangkat tersebut.

Contoh termasuk perangkat lunak dalam perangkat elektronik seperti oven microwave, router, atau kendaraan.

Jenis Perangkat Lunak

6. Perangkat Lunak Gratis dan Sumber Terbuka (Freeware and Open Source Software)

- Freeware: Perangkat lunak yang didistribusikan secara gratis namun tetap memiliki hak cipta. Pengguna dapat menggunakannya tanpa biaya tetapi tidak dapat memodifikasi kode sumbernya. Contoh: Adobe Acrobat Reader, Skype.
- Open Source Software: Perangkat lunak yang kode sumbernya tersedia untuk umum, sehingga siapa saja dapat memodifikasi, memperbaiki, dan mendistribusikannya. Contoh: Linux, Mozilla Firefox, LibreOffice.

Jenis Perangkat Lunak

7. Perangkat Lunak Komersial (Commercial Software)

Perangkat lunak yang dijual atau dilisensikan untuk menghasilkan keuntungan. Biasanya, perangkat lunak ini memiliki batasan lisensi yang ketat. Contoh: Microsoft Office, Adobe Photoshop.

Jenis Perangkat Lunak

8. Perangkat Lunak Percobaan (Shareware)

Perangkat lunak yang dapat digunakan secara gratis untuk jangka waktu tertentu atau dengan fitur terbatas. Setelah masa percobaan berakhir, pengguna harus membayar untuk terus menggunakannya atau untuk mengakses fitur lengkapnya. Contoh: WinRAR, beberapa versi antivirus.

Jenis Perangkat Lunak

9. Perangkat Lunak Adware (Ad-supported Software)

Perangkat lunak yang didistribusikan secara gratis, namun menampilkan iklan kepada pengguna. Pendapatan dari iklan membantu pembuat perangkat lunak menutupi biaya pengembangan. Contoh: Beberapa aplikasi seluler atau browser toolbar.

Jenis Perangkat Lunak

10. Perangkat Lunak Malware (Malicious Software)

Perangkat lunak berbahaya yang dirancang untuk merusak, mengganggu, atau mendapatkan akses yang tidak sah ke sistem komputer. Jenis ini mencakup virus, worm, ransomware, spyware, dan trojan horse.

Sifat Perangkat Lunak

- Abstract
  - Perangkat lunak yang bersifat abstrak merujuk pada kenyataan bahwa perangkat lunak itu sendiri tidak memiliki wujud fisik seperti perangkat keras.
  - Karena sifatnya yang abstrak, perangkat lunak bisa dengan mudah digandakan, didistribusikan, dan dimodifikasi tanpa perlu melibatkan perubahan fisik.
- Intangible
  - Perangkat lunak (software) yang tidak memiliki wujud fisik disebut sebagai "aset intangible" atau "software intangible." Ini berarti bahwa perangkat lunak tersebut tidak dapat disentuh atau dilihat secara langsung seperti benda fisik pada umumnya.
  - Keberadaannya tergantung pada kode-kode komputer dan instruksi yang menjalankan fungsi-fungsinya.

Faktor kegagalan pengembangan perangkat lunak

- Permintaan yang tinggi
  - Permintaan atau harapan dari pengguna, pemangku kepentingan, atau pasar terhadap perangkat lunak jauh melebihi kapasitas tim pengembang untuk memenuhinya.
  - Hal ini termasuk: tuntutan fitur yang berlebihan, tekanan untuk cepat rilis, ketidakmampuan untuk skala, keterbatasan sumber daya.
- Tidak menerapkan teknik dalam *software engineering*

Rekayasa Perangkat Lunak (*Software Engineering*)

- Teknik rekayasa yang berfokus pada seluruh aspek pengembangan perangkat lunak mulai dari spesifikasi kebutuhan hingga perawatan (*maintenance*) setelah digunakan.
- Disiplin ilmu yang fokus pada penerapan pendekatan sistematis, terstruktur, dan terukur untuk pengembangan, operasi, dan pemeliharaan perangkat lunak.
- Tujuan utama rekayasa perangkat lunak adalah menghasilkan perangkat lunak yang berkualitas tinggi, yang dapat diandalkan, efisien, aman, dan dapat dipelihara, sambil memenuhi kebutuhan pengguna dan batasan anggaran serta waktu.

Karakteristik perangkat lunak yang baik

- *Maintainable*

  Perangkat lunak harus ditulis sedemikian rupa sehingga dapat berkembang untuk memenuhi perubahan kebutuhan pelanggan.

- *Secure and dependable* (ketergantungan)

  Ketergantungan perangkat lunak mencakup serangkaian karakteristik termasuk keandalan (reliability), keamanan (security), dan keselamatan (safety). Perangkat lunak yang dapat diandalkan tidak boleh menyebabkan kerusakan fisik atau ekonomi jika terjadi kegagalan sistem.

- *Efficient*

  Perangkat lunak tidak boleh membuang-buang sumber daya sistem seperti memori dan siklus prosesor. Ffisiensi mencakup daya tanggap (responsiveness), waktu pemrosesan, pemanfaatan memori, dll.

- *Acceptable*

  Harus dapat dimengerti (understandble), dapat digunakan (usable), dan kompatibel dengan sistem lain yang mereka gunakan.

Kenapa *software engineering* penting?

- Ketergantungan kepada *software* yang tinggi
- Jika diterapkan dapat menekan biaya.
    - Hampir 60% biaya software berupa biaya pengembangan (development costs), 40% biaya pengujian (testing costs).
    - Untuk software custom (berdasarkan permintaan khusus), biaya evolusi (evolution costs) biasanya melebihi development costs.

Etika dalam *software engineering*

- *Confidentiality*

  Harus menghormati kerahasiaan perusahaan atau klien.

- *Competence*

  Tidak boleh secara sadar menerima pekerjaan di luar kompetensi.

- *Intellectual property rights*

  Mengetahui undang-undang setempat yang mengatur penggunaan kekayaan intelektual seperti paten dan hak cipta, memastikan bahwa kekayaan intelektual pemberi kerja dan klien dilindungi.

- *Computer misuse*

  Penyalahgunaan komputer berkisar dari hal yang sepele (misalnya bermain game di komputer perusahaan) hingga yang sangat serius (penyebaran virus atau malware lainnya).

Baca kode etik *software engineering* ACM/IEEE tahun 1999

Studi Kasus

Kegagalan Sistem Denver International Airport (DIA) Baggage Handling

- Sistem penanganan bagasi otomatis di Bandara Internasional Denver adalah salah satu proyek perangkat lunak yang paling ambisius pada masanya. Sistem ini dirancang untuk menangani bagasi secara otomatis dan efisien di seluruh bandara.
- Proyek ini mengalami penundaan selama hampir dua tahun, dengan anggaran yang membengkak dari $193 juta menjadi $560 juta.
- Masalah utamanya adalah kompleksitas sistem, perubahan spesifikasi yang konstan, kurangnya koordinasi antara tim pengembangan perangkat lunak, dan kegagalan dalam pengujian yang memadai.
- Manajemen Proyek: Kegagalan dalam perencanaan dan manajemen proyek menyebabkan penundaan dan pembengkakan biaya.
- Pengujian yang Tidak Memadai: Kurangnya pengujian menyeluruh sebelum peluncuran menyebabkan kegagalan sistem yang parah ketika diuji dalam kondisi nyata.
- Analisis Kebutuhan yang Buruk: Perubahan spesifikasi yang konstan menunjukkan bahwa analisis kebutuhan awal tidak dilakukan dengan baik.

Studi Kasus

Transformasi Digital di Netflix

- Netflix awalnya adalah perusahaan penyewaan DVD melalui pos, tetapi kemudian beralih ke platform streaming digital. Transformasi ini melibatkan pembuatan platform perangkat lunak yang dapat mengelola streaming video berkualitas tinggi ke jutaan pengguna di seluruh dunia.
- Netflix berhasil membangun arsitektur perangkat lunak berbasis cloud yang sangat skalabel dan andal, dengan kemampuan untuk memberikan pengalaman streaming yang mulus bahkan dengan permintaan pengguna yang sangat tinggi.
- Netflix menerapkan prinsip-prinsip rekayasa perangkat lunak untuk menciptakan sistem yang skalabel dan dapat menangani peningkatan jumlah pengguna dengan kinerja yang konsisten.
- Netflix menggunakan pendekatan DevOps dan CI/CD untuk memastikan bahwa perubahan perangkat lunak dapat diuji dan diterapkan dengan cepat, tanpa mengganggu layanan.
- Netflix mengadopsi arsitektur microservices, yang memungkinkan pengembangan, pengujian, dan pemeliharaan komponen perangkat lunak secara independen, meningkatkan fleksibilitas dan efisiensi.

Any questions?