Rencana Pembelajaran Semester (RPS) Rekayasa Perangkat Lunak

Program Studi: S1 Informatika
Semester: 4
Bobot SKS: 3 SKS

Dosen Pengampu: Subur Anugerah, S.T., M.Eng.

A. Deskripsi Mata Kuliah

Mata kuliah ini membekali mahasiswa dengan pengetahuan dan keterampilan praktis dalam mengembangkan perangkat lunak berkualitas tinggi. Mahasiswa akan mempelajari prinsip-prinsip rekayasa perangkat lunak, metodologi pengembangan, serta teknik-teknik untuk setiap fase dalam siklus hidup perangkat lunak. Mata kuliah ini juga akan mengintegrasikan perkembangan terkini dalam bidang AI dan bagaimana AI dapat digunakan untuk meningkatkan proses rekayasa perangkat lunak serta membangun sistem cerdas.

B. Capaian Pembelajaran (*Learning Outcomes*)

Setelah mengikuti mata kuliah ini, mahasiswa diharapkan mampu:

1. **Memahami Prinsip Dasar:** Memahami prinsip-prinsip dasar rekayasa perangkat lunak, etika profesi, dan pentingnya kualitas perangkat lunak.
2. **Menerapkan Metodologi:** Menerapkan metodologi pengembangan perangkat lunak yang sesuai (misalnya, Waterfall, Agile, Spiral) dengan mempertimbangkan kelebihan dan kekurangannya.
3. **Analisis Kebutuhan:** Melakukan analisis kebutuhan perangkat lunak secara komprehensif, termasuk *user requirements gathering*, dan mendokumentasikannya dengan baik.
4. **Desain Perangkat Lunak:** Merancang arsitektur perangkat lunak, antarmuka pengguna (UI/UX), dan basis data yang sesuai dengan kebutuhan.
5. **Pengkodean:** Menerapkan prinsip-prinsip pengkodean yang baik, bersih, terdokumentasi, dan mudah dipelihara.

6. **Pengujian Perangkat Lunak:** Melakukan pengujian perangkat lunak secara menyeluruh (unit testing, integration testing, system testing, user acceptance testing) untuk memastikan kualitas.
7. **Manajemen Proyek:** Mengelola proyek perangkat lunak secara efektif, termasuk perencanaan, penjadwalan, estimasi biaya, dan manajemen risiko.
8. **Memahami Peran AI:** Memahami peran AI dalam rekayasa perangkat lunak, baik sebagai alat bantu maupun sebagai komponen dalam sistem yang dikembangkan (misalnya, *AI-powered testing*, *intelligent code completion*, sistem rekomendasi, dll.).
9. **Data Migration dan Go-Live**: Memahami pentingnya strategi migrasi data dari sistem lama ke sistem baru, serta persiapan dan pelaksanaan *go-live* yang sukses.
10. **Etika dan Tanggung Jawab AI:** Memahami implikasi etika dan tanggung jawab sosial dalam pengembangan dan penggunaan AI dalam perangkat lunak.

C. Materi Pembelajaran (Pokok Bahasan)

Berikut adalah contoh materi pembelajaran yang dapat disesuaikan dengan kebutuhan dan perkembangan AI:

1. **Pengantar Rekayasa Perangkat Lunak:**
    - Definisi, tujuan, dan prinsip-prinsip rekayasa perangkat lunak.
    - Siklus hidup perangkat lunak (SDLC) dan berbagai modelnya (Waterfall, Agile, Spiral, dll.).
    - Etika profesi dan tanggung jawab *software engineer*.
    - Peran AI dalam rekayasa perangkat lunak:
        - AI sebagai alat bantu pengembangan (contoh: *GitHub Copilot*, *Tabnine*).
        - AI sebagai bagian dari perangkat lunak (contoh: *machine learning* untuk prediksi, *natural language processing* untuk chatbot).
2. **Software Requirements Engineering (dengan AI):**
    - ***User Requirements Gathering***:

        - Teknik elisitasi kebutuhan: wawancara, survei, *focus group discussion*, *prototyping*.
        - *Use case diagrams*, *user stories*, *user journey maps*.
        - Pemanfaatan AI untuk analisis sentimen dari *feedback* pengguna.
    - **Analisis Kebutuhan:**
        - Klasifikasi kebutuhan (fungsional, non-fungsional, *domain requirements*).
        - Pemodelan kebutuhan: *Data Flow Diagrams* (DFD), *Entity-Relationship Diagrams* (ERD).
        - Prioritisasi kebutuhan (MoSCoW, Kano Model).
        - Validasi dan verifikasi kebutuhan.
        - *AI-powered requirements analysis*: penggunaan NLP untuk mengidentifikasi kebutuhan yang ambigu atau tidak konsisten.
3. **Software Design (dengan AI):**
    - Prinsip-prinsip desain: *modularity, cohesion, coupling, abstraction, information hiding*.
    - Desain arsitektur perangkat lunak: *layered architecture, microservices architecture, client-server, MVC*.
    - Desain antarmuka pengguna (UI/UX): *usability, accessibility, user-centered design*.
    - Desain basis data: *relational database design, NoSQL databases*.
    - *AI-powered design*: penggunaan AI untuk menghasilkan *mockup* UI/UX, rekomendasi arsitektur, atau optimasi desain basis data.
4. **Software Construction (Pengkodean dengan AI):**
    - Pemilihan bahasa pemrograman dan *framework*.
    - Prinsip-prinsip pengkodean yang baik: *clean code, code style, documentation*.
    - *Version control* dengan Git dan GitHub.
    - *Code review* dan *pair programming*.
    - *AI-powered coding*:
        - *Intelligent code completion* (contoh: *GitHub Copilot*, *Tabnine*).
        - *Automated code refactoring*.

        - *Bug detection* dan *prediction* dengan AI.
5. **Software Testing (dengan AI):**
    - Jenis-jenis pengujian:
        - *Unit testing*
        - *Integration testing*
        - *System testing*
        - *User Acceptance Testing* (UAT)
        - Pengujian Non-Fungsional (Performa, Keamanan, Kegunaan, dll.)
    - Teknik pengujian: *black-box testing, white-box testing, grey-box testing*.
    - Otomatisasi pengujian.
    - *AI-powered testing*:
        - *Test case generation* otomatis.
        - *Visual testing* dengan AI.
        - *Self-healing tests*.
6. **Data Migration:**
    - Perencanaan migrasi data
    - Strategi migrasi data (Big Bang, Trickle, dll)
    - Pemetaan data (Data mapping)
    - Validasi dan rekonsiliasi data
    - Rollback plan
7. **Go-Live:**
    - Persiapan infrastruktur
    - Pelatihan pengguna
    - Cutover planning
    - Monitoring dan dukungan pasca-implementasi
8. **Software Maintenance and Evolution:**
    - Jenis-jenis pemeliharaan: *corrective, adaptive, perfective, preventive*.
    - *Legacy systems* dan *reverse engineering*.
    - *Refactoring* dan *reengineering*.
9. **Software Project Management:**
    - Perencanaan proyek: *Work Breakdown Structure* (WBS), *Gantt chart*.
    - Estimasi biaya dan jadwal.
    - Manajemen risiko.

    - Manajemen tim dan komunikasi.
    - Metodologi Agile dan Scrum.
10. **Topik Khusus (Opsional):**
    - *DevOps* dan *Continuous Integration/Continuous Delivery* (CI/CD).
    - *Cloud computing* untuk pengembangan perangkat lunak.
    - Keamanan perangkat lunak (*secure coding practices*, *penetration testing*).
    - Aspek hukum dan etika dalam pengembangan perangkat lunak.
    - *AI Ethics* dan *Responsible AI*.

D. Penilaian

- **Tugas Individu (30%):** Latihan soal, *coding assignments*, *mini-projects*.
- **Tugas Kelompok (30%):** Studi kasus, proyek pengembangan perangkat lunak skala kecil (dengan penerapan AI jika memungkinkan).
- **Ujian Tengah Semester (UTS) (20%):** Tes tertulis atau *take-home exam*.
- **Ujian Akhir Semester (UAS) (20%):** Tes tertulis, presentasi proyek, atau demonstrasi perangkat lunak.

E. Referensi

- **Utama:**
    - Sommerville, Ian. *Software Engineering*. Edisi terbaru.
    - Pressman, Roger S. *Software Engineering: A Practitioner's Approach*. Edisi terbaru.
    - Buku-buku dan artikel ilmiah tentang AI dalam rekayasa perangkat lunak.

- **Pendukung:**
    - Materi online (tutorial, video, dokumentasi) tentang *tools* dan *framework* yang relevan.
    - *Case studies* tentang penerapan AI dalam proyek-proyek perangkat lunak.

- Jurnal: IEEE Transactions on Software Engineering, ACM Transactions on Software Engineering and Methodology

F. Jadwal Pembelajaran

Jadwal pembelajaran

Minggu	Topik	Aktivitas
1	Pengantar Rekayasa Perangkat Lunak	Kuliah, diskusi, pengenalan *tools*
2	*Software Requirements Engineering* (1)	Kuliah, studi kasus, latihan *user requirements gathering*
3	*Software Requirements Engineering* (2)	Kuliah, workshop analisis kebutuhan, *brainstorming* ide proyek kelompok
4	*Software Design* (1)	Kuliah, latihan desain arsitektur, diskusi kelompok tentang desain proyek
5	*Software Design* (2)	Kuliah, workshop desain UI/UX, presentasi *progress* desain proyek
6	*Software Construction* (1)	Kuliah, *live coding*, pengenalan Git dan GitHub
7	*Software Construction* (2)	Kuliah, *code review*, diskusi tentang *best practices* dalam pengkodean, *pair programming*
8	**UTS**	Ujian Tengah Semester
9	*Software Testing* (1)	Kuliah, workshop *unit testing*, diskusi kelompok tentang strategi pengujian

Minggu	Topik	Aktivitas
10	*Software Testing* (2)	Kuliah, demonstrasi *automated testing*, presentasi *progress* pengujian proyek
11	Data Migration	Kuliah, studi kasus, perencanaan data migration
12	Go-Live	Kuliah, simulasi go-live, diskusi tentang tantangan dan solusi
13	*Software Maintenance and Evolution*	Kuliah, diskusi tentang *legacy systems*, latihan *refactoring*
14	*Software Project Management*	Kuliah, studi kasus, simulasi manajemen proyek
15	Topik Khusus / Presentasi Proyek Akhir	Kuliah tamu, presentasi proyek akhir, demonstrasi perangkat lunak
16	**UAS**	Ujian Akhir Semester

G. Catatan Tambahan

- **Proyek Kelompok:** Mahasiswa akan dibagi menjadi kelompok-kelompok kecil untuk mengerjakan proyek pengembangan perangkat lunak. Proyek ini akan mencakup semua tahapan SDLC, dan mahasiswa didorong untuk menerapkan konsep-konsep AI yang relevan.
- **Penggunaan *Tools*:** Mahasiswa akan diperkenalkan dengan berbagai *tools* yang digunakan dalam rekayasa perangkat lunak, seperti:
    - *Version control*: Git, GitHub
    - IDE: VS Code, IntelliJ IDEA, PyCharm
    - *Project management*: Jira, Trello
    - *Testing*: JUnit, Selenium
    - *AI-powered tools*: GitHub Copilot, Tabnine

- **Fleksibilitas:** RPS ini bersifat fleksibel dan dapat disesuaikan dengan kebutuhan, perkembangan teknologi, dan *feedback* dari mahasiswa.
- **Keterlibatan Aktif:** Mahasiswa didorong untuk aktif berpartisipasi dalam diskusi, bertanya, dan berbagi pengalaman.
- **Studi Kasus Nyata:** Gunakan studi kasus dari perusahaan teknologi ternama (misalnya, Google, Microsoft, Amazon) tentang bagaimana mereka menerapkan AI dalam pengembangan perangkat lunak.