Nama:Muhamad Farras Jasir

NIM:312210361

Kelas:TI.22.A5

1. Rekayasa perangkat lunak (software engineering) adalah disiplin ilmu yang berkaitan dengan proses pembuatan, pengembangan, pemeliharaan, dan evolusi perangkat lunak secara sistematis dan terkendali. Tujuan utamanya adalah untuk menghasilkan perangkat lunak yang berkualitas tinggi, handal, efisien, dan dapat dipertahankan.

   Sejarah Rekayasa Perangkat Lunak: Rekayasa perangkat lunak sebagai disiplin ilmu mulai diperkenalkan pada tahun 1960-an ketika industri perangkat lunak mulai berkembang pesat. Pada awalnya, proses pengembangan perangkat lunak dilakukan secara ad hoc dan tidak terstruktur. Namun, dengan semakin kompleksnya perangkat lunak yang dibuat, diperlukan pendekatan yang lebih sistematis dan terkoordinasi.

   Metode Rekayasa Perangkat Lunak: Ada beberapa metode yang digunakan dalam rekayasa perangkat lunak, di antaranya:

Model Air Terjun (Waterfall Model): Model ini menggambarkan proses pengembangan perangkat lunak sebagai serangkaian tahapan linear, mulai dari analisis, perancangan, implementasi, pengujian, hingga pemeliharaan. Setiap tahap dilakukan secara berurutan dan seringkali tidak ada kemungkinan kembali ke tahap sebelumnya.

Model Spiral: Model ini menggabungkan pendekatan iteratif dengan pengelolaan risiko. Proses pengembangan perangkat lunak dilakukan dalam siklus berulang yang terdiri dari tahap perencanaan, analisis risiko, pengembangan, dan evaluasi. Setiap siklus spiral memperluas fungsionalitas perangkat lunak.

Metode Pengembangan Berbasis Komponen: Pendekatan ini melibatkan penggunaan komponen perangkat lunak yang dapat digunakan kembali (reusable) untuk mempercepat proses pengembangan. Komponen-komponen ini dibangun secara terpisah dan kemudian diintegrasikan ke dalam perangkat lunak yang sedang dikembangkan.

Perkembangan Rekayasa Perangkat Lunak: Rekayasa perangkat lunak terus mengalami perkembangan seiring waktu. Beberapa tren dan perkembangan penting dalam industri rekayasa perangkat lunak meliputi:

Metode Pengembangan Agil: Metode ini menekankan kolaborasi tim, fleksibilitas, dan responsif terhadap perubahan yang cepat. Beberapa metode pengembangan agil yang populer termasuk Scrum, Extreme Programming (XP), dan Kanban.

DevOps: DevOps adalah pendekatan yang menggabungkan pengembangan perangkat lunak (Development) dengan operasi infrastruktur TI (Operations). DevOps bertujuan untuk meningkatkan kerjasama antara tim pengembangan dan operasi, mempercepat siklus pengembangan, dan meningkatkan kualitas perangkat lunak.

Rekayasa Perangkat Lunak Terkelola (Managed Software Engineering): Pendekatan ini melibatkan penggunaan platform dan alat bantu otomatis untuk mengelola siklus hidup perangkat lunak, termasuk pemeliharaan, pemantauan, dan peningkatan kinerja.

Kualitas Perangkat Lunak: Penekanan pada aspek kualitas perangkat lunak semakin meningkat. Beberapa pendekatan seperti Tes Otomatis, Integrasi Berkelanjutan (Continuous Integration), dan Pengujian Fungsional Terotomatisasi (Automated Functional Testing) digunakan untuk memastikan bahwa perangkat lunak memenuhi persyaratan fungsional dan nonfungsional.

Perkembangan teknologi dan kebutuhan bisnis yang terus berkembang terus mendorong inovasi dan perkembangan dalam rekayasa perangkat lunak.

2. Analisis desain pendekatan terstruktur (Structured Design) adalah metodologi yang digunakan dalam rekayasa perangkat lunak untuk merancang sistem dengan pendekatan yang terstruktur dan sistematis. Pendekatan ini berfokus pada pemisahan masalah menjadi bagian-bagian yang lebih kecil, yang kemudian diorganisasikan secara hierarkis dan saling terkait.

   Tujuan utama dari analisis desain pendekatan terstruktur adalah untuk menghasilkan desain yang mudah dimengerti, mudah dipelihara, dan dapat diimplementasikan dengan efisien. Pendekatan ini melibatkan beberapa konsep dan teknik yang terintegrasi dalam proses analisis dan desain sistem.

Berikut adalah langkah-langkah umum dalam analisis desain pendekatan terstruktur:

Penentuan Persyaratan: Langkah pertama adalah memahami persyaratan sistem dengan mendefinisikan kebutuhan pengguna, fungsi yang diinginkan, dan batasan sistem.

Analisis Sistem: Proses ini melibatkan pemahaman yang mendalam tentang sistem yang sedang dianalisis. Ini melibatkan identifikasi fungsi-fungsi sistem, aliran data, dan interaksi antara komponen sistem.

Pembuatan Diagram Konteks: Diagram konteks digunakan untuk menggambarkan hubungan antara sistem yang sedang dianalisis dengan entitas eksternal yang berinteraksi dengan sistem. Ini membantu dalam memahami peran dan batasan sistem.

Dekomposisi: Sistem yang kompleks dibagi menjadi modul-modul yang lebih kecil yang lebih mudah dikelola. Proses ini melibatkan identifikasi modul utama dan pemisahan fungsi-fungsi sistem menjadi unit yang lebih terdefinisi.

Pembuatan Diagram Hierarchy: Diagram hierarki digunakan untuk menggambarkan hubungan hierarkis antara modul-modul dalam sistem. Diagram ini menunjukkan bagaimana modul-modul saling terkait dan berinteraksi.

Penentuan Antarmuka: Setiap modul dianalisis dan antarmuka antara modul-modul ditentukan. Hal ini meliputi definisi masukan, keluaran, dan prosedur yang digunakan untuk berkomunikasi antara modul-modul.

Perancangan Algoritma: Proses ini melibatkan perancangan algoritma untuk setiap modul yang menjelaskan langkah-langkah logis yang diperlukan untuk mencapai fungsi yang diinginkan.

Validasi Desain: Desain yang dihasilkan dievaluasi dan divalidasi untuk memastikan bahwa desain tersebut memenuhi persyaratan sistem dan memecahkan masalah yang ada.

Keuntungan dari analisis desain pendekatan terstruktur termasuk pemahaman yang lebih baik tentang sistem, kemudahan pemeliharaan, dan modularitas yang memungkinkan pengembangan paralel. Pendekatan ini juga membantu dalam mengurangi kompleksitas dan meningkatkan keandalan sistem.

Namun, analisis desain pendekatan terstruktur memiliki kelemahan dalam menghadapi sistem yang sangat kompleks dan sulit untuk mengikuti perubahan yang cepat. Dalam beberapa kasus, pendekatan yang lebih fleksibel seperti metode pengembangan berbasis komponen atau metode pengembangan berbasis objek mungkin lebih sesuai.

3. Analisis desain berorientasi objek (Object-Oriented Design) adalah pendekatan dalam rekayasa perangkat lunak yang berfokus pada pemodelan sistem menggunakan konsep objek. Pendekatan ini memungkinkan pemisahan masalah menjadi entitas yang lebih kecil yang disebut objek, yang memiliki atribut dan perilaku tertentu.

   Pada analisis desain berorientasi objek, sistem dilihat sebagai kumpulan objek yang saling berinteraksi untuk mencapai tujuan. Objek dapat merepresentasikan entitas nyata atau konsep abstrak dalam sistem yang sedang dianalisis. Masing-masing objek memiliki atribut (data) dan metode (fungsi) yang berhubungan dengan objek tersebut.

Berikut adalah konsep utama dalam analisis desain berorientasi objek:

Enkapsulasi: Enkapsulasi adalah konsep yang memungkinkan objek untuk menyembunyikan detail internalnya dan hanya mengekspos antarmuka publik. Objek berkomunikasi satu sama lain melalui antarmuka publik mereka. Hal ini membantu dalam memisahkan perhatian dan memungkinkan perubahan internal tanpa mempengaruhi objek lain.

Pewarisan (Inheritance): Pewarisan adalah konsep di mana objek baru dapat mewarisi properti dan perilaku dari objek yang sudah ada. Objek baru yang mewarisi disebut kelas turunan atau subclass, sedangkan objek yang memberikan warisan disebut kelas induk atau superclass. Konsep ini memungkinkan penggunaan kembali kode, memfasilitasi hierarki kelas, dan memungkinkan polimorfisme.

Polimorfisme: Polimorfisme adalah konsep di mana objek memiliki banyak bentuk. Objek dari kelas yang berbeda dapat merespons secara berbeda terhadap metode yang sama. Hal ini memungkinkan penggunaan metode generik yang dapat diterapkan pada objek dengan perilaku yang berbeda.

Asosiasi: Asosiasi adalah hubungan antara dua atau lebih objek yang bekerja bersama untuk mencapai tujuan tertentu. Asosiasi dapat memiliki arah dan multiplicitas, dan dapat digambarkan dengan bantuan diagram kelas atau diagram hubungan.

Agregasi dan Komposisi: Agregasi dan komposisi adalah bentuk hubungan khusus antara objek. Agregasi menggambarkan hubungan "bagian dari" antara objek, di mana objek-objek yang terkait dapat ada secara independen. Komposisi adalah bentuk hubungan yang lebih ketat, di mana objek yang terkait tidak dapat ada secara independen dan menjadi bagian integral dari objek lain.

Abstraksi: Abstraksi adalah proses menyederhanakan dan memfokuskan pada informasi yang relevan dalam konteks sistem. Dalam analisis desain berorientasi objek, abstraksi melibatkan penentuan kelas-kelas, atribut, dan metode yang relevan untuk menggambarkan sistem.

Analisis desain berorientasi objek memungkinkan pengembangan sistem yang modular, fleksibel, dan mudah dimengerti. Pendekatan ini memfasilitasi pemodelan dunia nyata dalam perangkat lunak, memungkinkan penggunaan kembali kode yang efisien, dan meningkatkan keterbacaan dan pemeliharaan sistem.

4. UML (Unified Modeling Language) adalah bahasa standar yang digunakan untuk memodelkan, merancang, dan mendokumentasikan sistem perangkat lunak. UML menyediakan seperangkat notasi grafis yang digunakan untuk menggambarkan berbagai aspek sistem, termasuk struktur, perilaku, interaksi, dan komunikasi antar objek.

   Berikut adalah beberapa jenis diagram yang umum digunakan dalam UML:

   Diagram Kelas (Class Diagram): Diagram kelas digunakan untuk menggambarkan struktur statis dari sistem. Ini mengidentifikasi kelas-kelas yang ada dalam sistem, hubungan antara kelas-kelas tersebut, atribut dan metode yang dimiliki oleh setiap kelas, serta visibilitas dan tipe data dari atribut dan metode.

Diagram Aktivitas (Activity Diagram): Diagram aktivitas menggambarkan aliran kerja atau urutan aktivitas dalam sistem. Ini digunakan untuk menggambarkan proses bisnis, aliran kontrol, dan aktivitas yang dilakukan oleh objek dalam sistem.

Diagram Sekuensi (Sequence Diagram): Diagram sekuen menunjukkan interaksi antara objek dalam urutan waktu. Ini menggambarkan pesan-pesan yang dikirim antara objek-objek tersebut dan urutan eksekusi dari aktivitas dan metode.

Diagram Kasus Penggunaan (Use Case Diagram): Diagram kasus penggunaan membantu dalam memahami kebutuhan fungsional sistem dari perspektif pengguna. Ini menggambarkan interaksi antara aktor (entitas luar yang berinteraksi dengan sistem) dan kasus penggunaan (fungsi-fungsi sistem yang memberikan nilai kepada aktor).

Diagram Komponen (Component Diagram): Diagram komponen menggambarkan struktur fisik sistem dan hubungan antara komponen-komponen. Ini mengidentifikasi komponen-komponen perangkat lunak, pustaka, dan dependensi antara komponen-komponen tersebut.

Diagram Penyekalan (Deployment Diagram): Diagram penyekalan menggambarkan bagaimana komponen-komponen sistem didistribusikan dan diimplementasikan pada lingkungan fisik. Ini mencakup server, node, jaringan, dan hubungan fisik antara mereka.

UML memberikan cara yang jelas dan konsisten untuk menyajikan informasi tentang perangkat lunak. Notasi grafis yang didefinisikan dalam UML memungkinkan para pengembang perangkat lunak untuk berkomunikasi dan berkolaborasi secara efektif dalam merancang, memodelkan, dan membangun sistem perangkat lunak yang kompleks.

5. Program desain (design program) adalah langkah dalam proses pengembangan perangkat lunak di mana desain sistem atau komponen perangkat lunak dihasilkan. Ini melibatkan pemikiran dan analisis yang mendalam untuk merancang struktur, fungsi, dan interaksi dari perangkat lunak yang akan dikembangkan.

   Program desain berfokus pada bagaimana komponen perangkat lunak akan diimplementasikan, bagaimana komunikasi dan interaksi antara komponen akan terjadi, dan bagaimana struktur keseluruhan sistem akan dibangun. Tujuan dari program desain adalah menghasilkan desain yang efisien, mudah dimengerti, dan dapat diimplementasikan dengan baik.

   Beberapa hal yang perlu dipertimbangkan dalam program desain meliputi:

   Arsitektur Sistem: Ini melibatkan pembuatan rencana tingkat tinggi tentang bagaimana komponen-komponen perangkat lunak akan berinteraksi dan berkomunikasi satu sama lain. Arsitektur sistem menentukan struktur keseluruhan sistem dan menyediakan panduan bagi pengembang dalam mengimplementasikan komponen perangkat lunak.

   Desain Komponen: Komponen-komponen perangkat lunak individu dijelaskan dalam detail. Ini melibatkan pemikiran tentang tipe data yang akan digunakan, struktur data, algoritma yang diperlukan, dan metode yang akan diimplementasikan. Desain komponen juga mencakup pemikiran tentang bagaimana komponen tersebut akan berinteraksi dengan komponen lain dalam sistem.

Antarmuka Pengguna: Jika perangkat lunak memiliki antarmuka pengguna, program desain juga akan mencakup desain antarmuka pengguna. Ini melibatkan pemikiran tentang tata letak, elemen-elemen antarmuka, navigasi, dan interaksi antara pengguna dan sistem.

Pengelolaan Data: Desain program juga melibatkan pemikiran tentang bagaimana data akan dikelola dalam sistem. Ini mencakup pemikiran tentang struktur data yang tepat, pengelolaan basis data, operasi input/output data, dan kebijakan keamanan data.

Pengujian dan Pemeliharaan: Program desain juga harus mempertimbangkan pengujian dan pemeliharaan sistem. Ini melibatkan memikirkan tentang bagaimana menguji komponen dan sistem secara menyeluruh, serta bagaimana memudahkan pemeliharaan dan perbaikan jika ada masalah di kemudian hari.

Program desain sering menggunakan notasi grafis seperti diagram alir, diagram kelas, dan diagram sekuens untuk menggambarkan struktur dan interaksi dalam sistem perangkat lunak. Selain itu, program desain juga dapat melibatkan penggunaan alat bantu desain perangkat lunak yang membantu dalam merancang dan menganalisis sistem dengan lebih efisien.

Program desain merupakan langkah penting dalam pengembangan perangkat lunak karena membentuk dasar implementasi dan memastikan bahwa sistem yang dihasilkan memenuhi persyaratan dan spesifikasi yang telah ditetapkan.

6. pengembangan belajar berbasis web universitas pelita bangsa

7. Analisa Mahasiswa.

Tahap ini peneliti melakukan observasi untuk mengetahui karakteristik mahasiswa. Karakteristik ini meliputi usia dan kemampuan keterampilan mahasiswa. Usia mahasiswa berkisar antara 20-23 tahun. Perkembangan kognitif (kemampuan berpikir) pada usia tersebut dijelaskan seperti berikut ini 1) Secara intelektual remaja dapat mulai berpikit logis tentang gagasan abstrak. 2) Berfungsinya kegiatan kognitif tingkat tinggi yaitu membuat rencana, sttrategi, keputusan-keputusan, serta memecahkan masalah. 3) Munculnya kemampuan nalar secara ilmiah, belajar menguji hipotesis. 4) Memikirkan masa depan, perencanaan, dan mengeksplorasi alternative untuk mencapainya psikologi remaja. 5) Wawasan berfikirnya semakin meluas, bisa meliputi agama, keadilan, moralitas, dan identitas (jati diri) (Saputri and Hannah 2018). Berdasarkan analisis terhadap mahasiswa tersebut, dijadikan pertimbangan dalam pengembangan media web pada mata kuliah pemrograman web dasar. Pengembangan media pembelajaran berbasis web yang dikembangkan telah sesuai dengan kondisi dan karakteristik mahasiswa. Dengan adanya media ini diharapkan mahasiswa dapat lebih mudah memahami materi perkuliahan

A. Rumusan Masalah

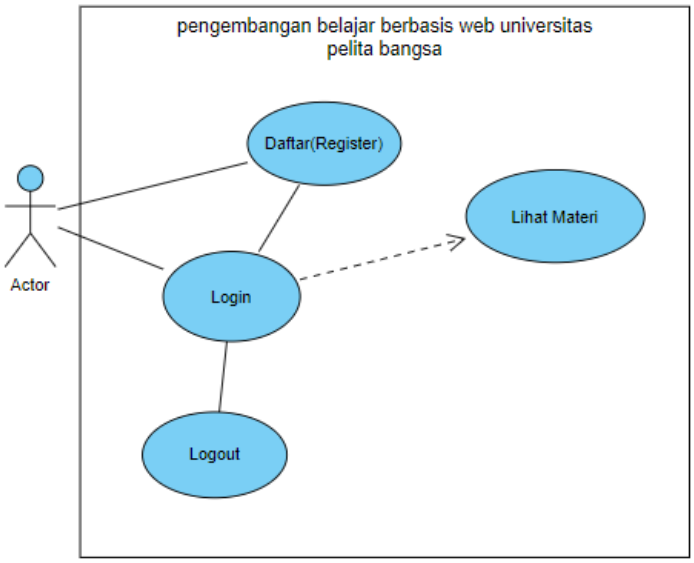
Berdasarkan hasil observasi yang dilakukan, peneliti mendapatkan beberapa informasi mengenai permasalahan yang terjadi selama kegiatan pembelajaran berlangsung. Kenyataannya sebagian mahasiswa masih sulit untuk meningkatkan keterampilan pembuatan website, karena waktu perkuliahan yang disediakan sangat terbatas. Media penunjang yang digunakan yaitu e-learning kampus belum mampu

membantu mahasiswa, hal ini karena fitur yang terdapat pada e-learning tersebut masih kurang. Salah satu alternatife media yang bisa digunakan adalah media pembelajaran berbasis web dengan fitur-fitur terbaru yang dikembangkan sesuai kebutuhan mahasiswa seperti adanya video, diskusi grup, simulasi pengkodean web, serta latihan. Berdasarkan uraian diatas maka penelitian ini dilaksanakan untuk mengembangkan media pembelajaran berbasis web terhadap keterampilan mahasiswa Prodi Pendidikan Teknik Informatika Universitas Pelita Bangsa pada mata kuliah pemrograman web dasar. Media pembelajaran berbasis web yang akan dikembangkan digunakan sebagai penunjang kegiatan perkuliahan baik dikelas maupun diluar kelas. Media pembelajaran berbasis web yang dihasilkan bersifat online serta memiliki fitur diskusi grup, simulasi dalam pengkodean web, dan dilengkapi dengan video-video materi pembelajaran. Selain itu, hasil penelitian ini dapat dijadikan sebagai referensi dalam mengembangkan media pembelajaran berbasis web.

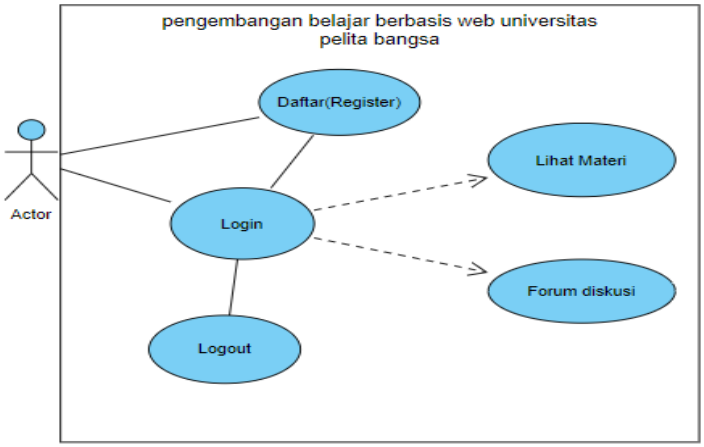
B. Kebutuhan Sistem

Pemilihan media. Media yang akan dikembangan yaitu media pembelajaran berbasis web yang berisi fitur video, forum diskusi realtime, simulasi praktek pengkodean website. b) Rancangan awal. Dibagi menjadi 2 tahapan yaitu, desain logis dan desain fisik. Hasil pembuatan desain logis ini adalah konteks diagram (context diagram), DFD (Data Flow Diagram) yang berfungsi untuk memudahkan pengguna memahami bagaimana media web yang dikembangkan berjalan serta mengetahui apa saja aktivitas yang dilakukan pengguna (Muhammad Hakiki 2021). Desain fisik bertujuan untuk merancang antarmuka atau tampilan pengguna dengan media web.c) Pembuatan media. Dilakukan untk memulai proses pembuatan media pembelajaran berbasis web dengan menggunakan aplikasi pendukung dan bahasa pemrograman. Aplikasi yang digunakan untuk membuat media pembelajaran berbasis web yaitu, Sublime Text, XAMPP, SQL, Adobe Photoshop, Bahasa pemrograman PHP, CodeIgniter dan JavaScrip

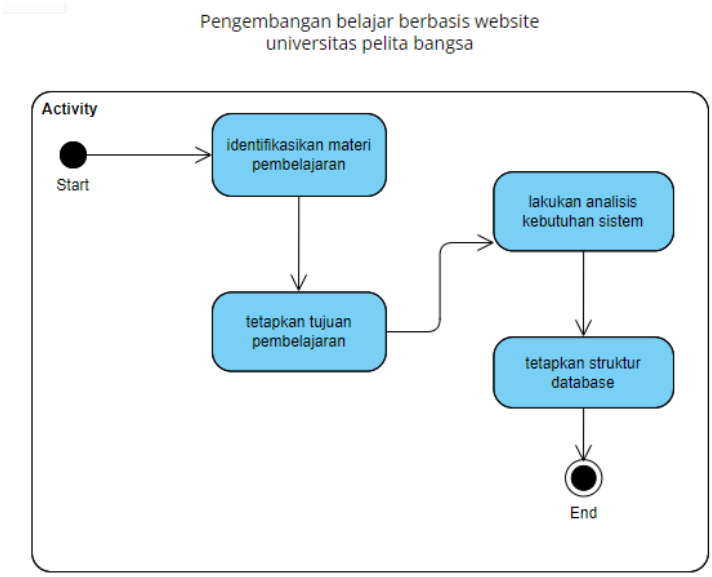
C. Use Case Diagram



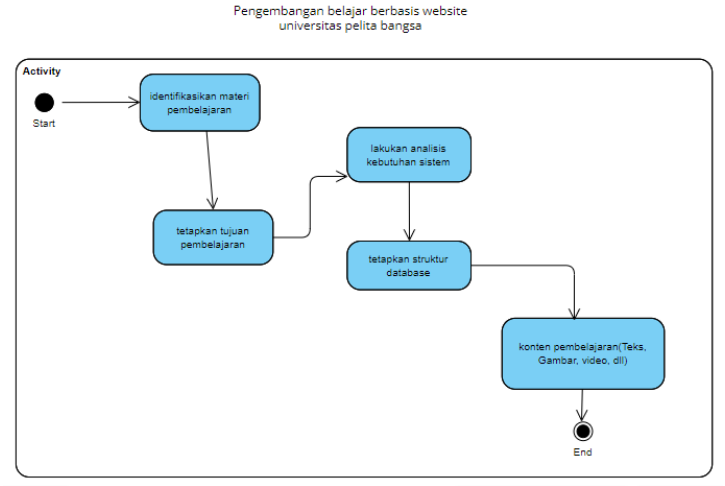
Use Case Diagram(Alternatif)



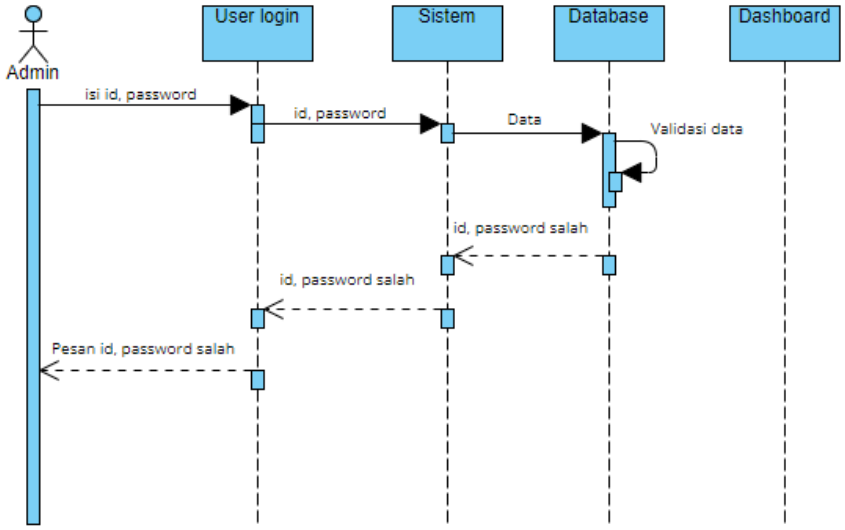
D. Activity Diagram



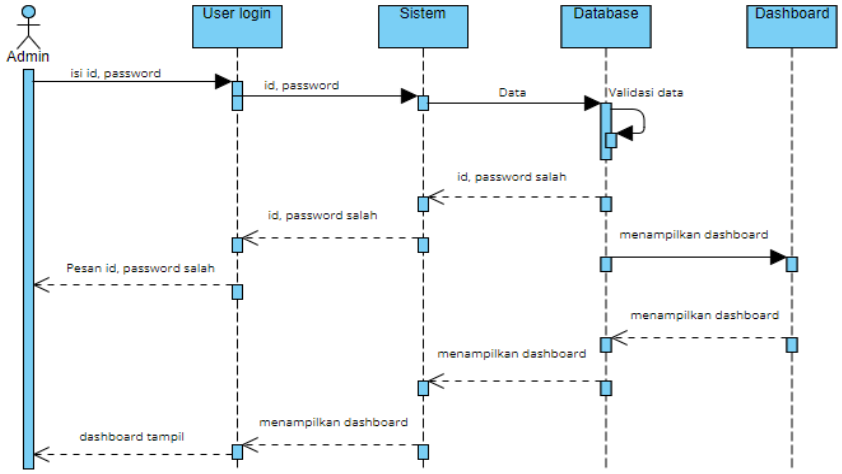
Activity Diagram (Alternatif)



E. Squenchy Diagram

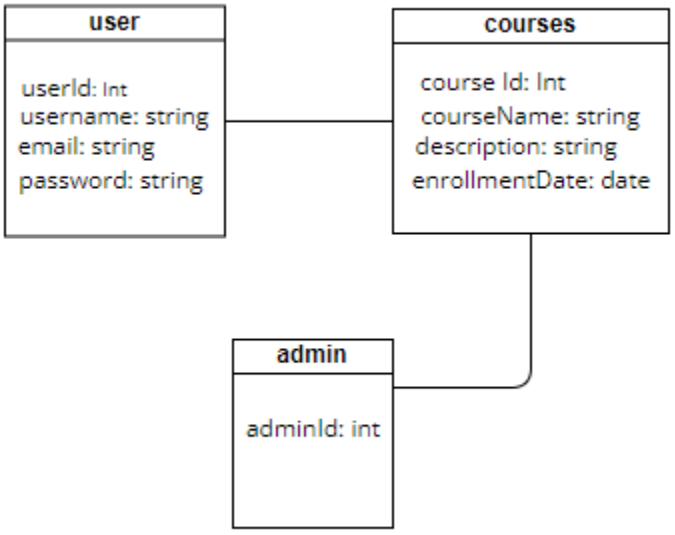


Squenchy Diagram (Alternatif)

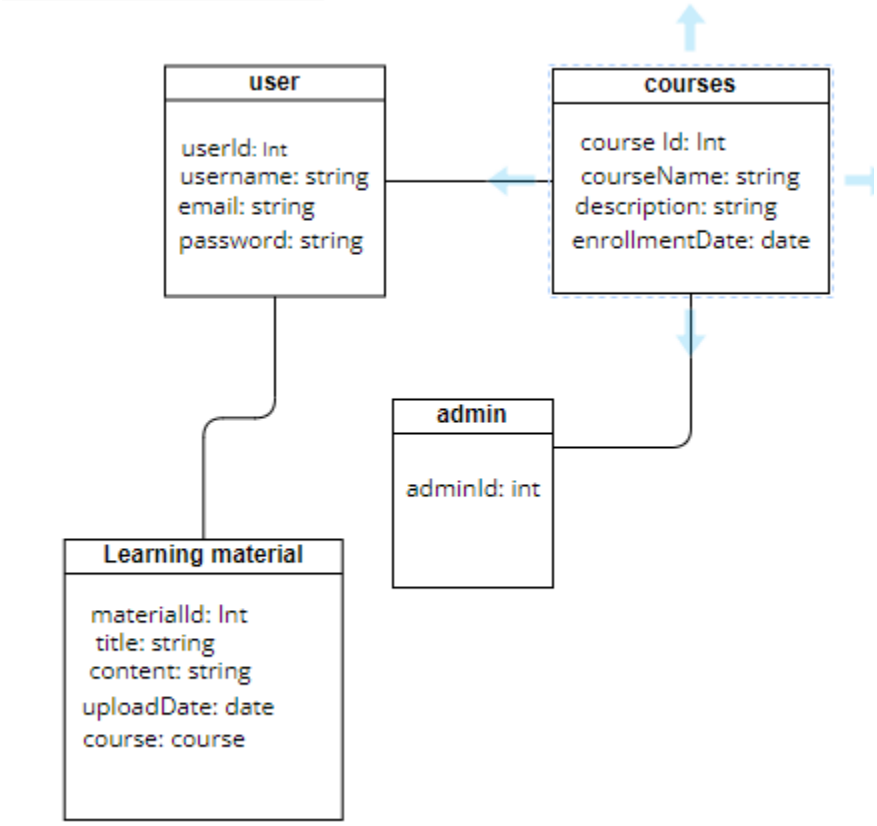


BagiaN II

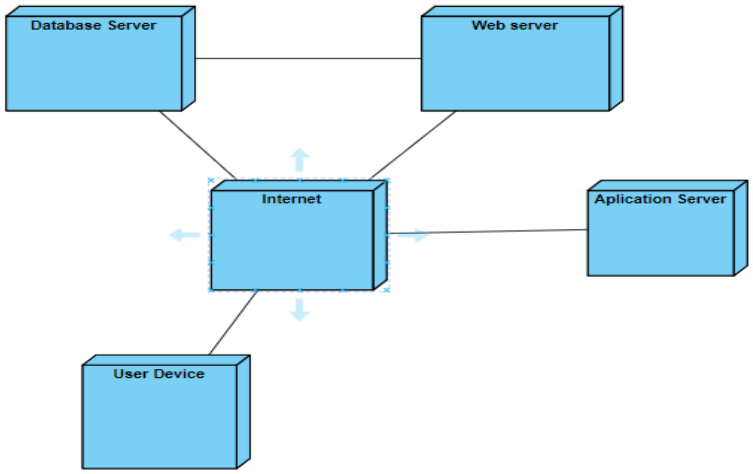
A. Class Diagram



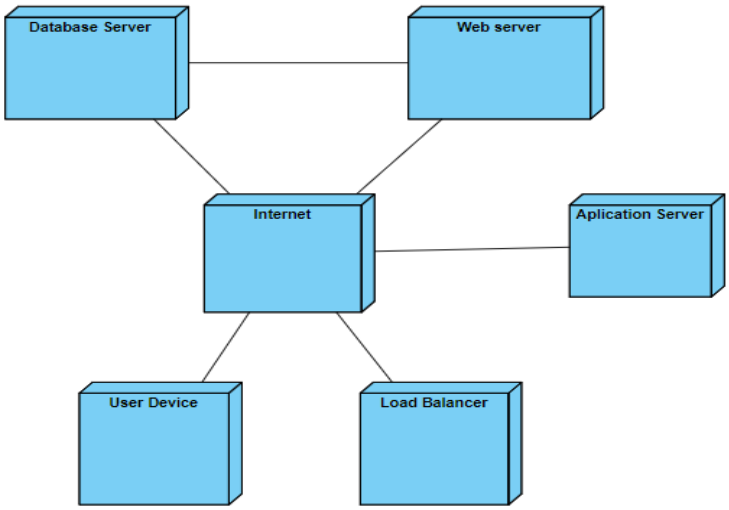
Class diagram (Alternatif)



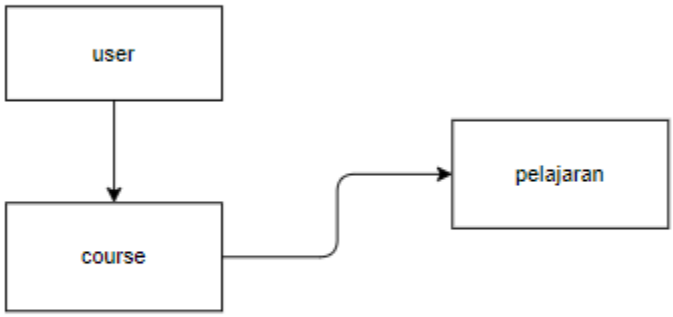
B. Deployment Diagram



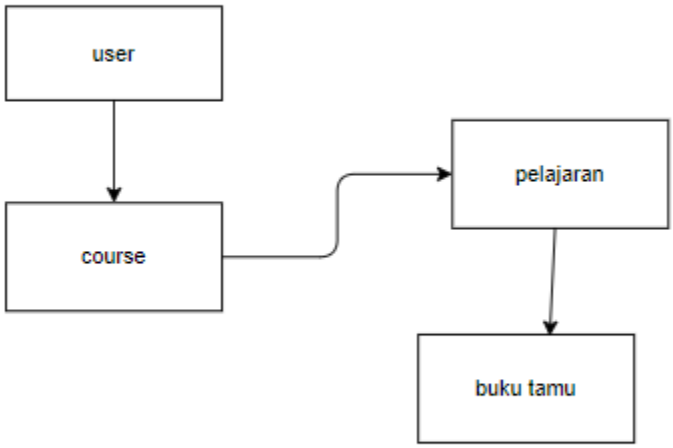
Deployment Diagram (Alternatif)



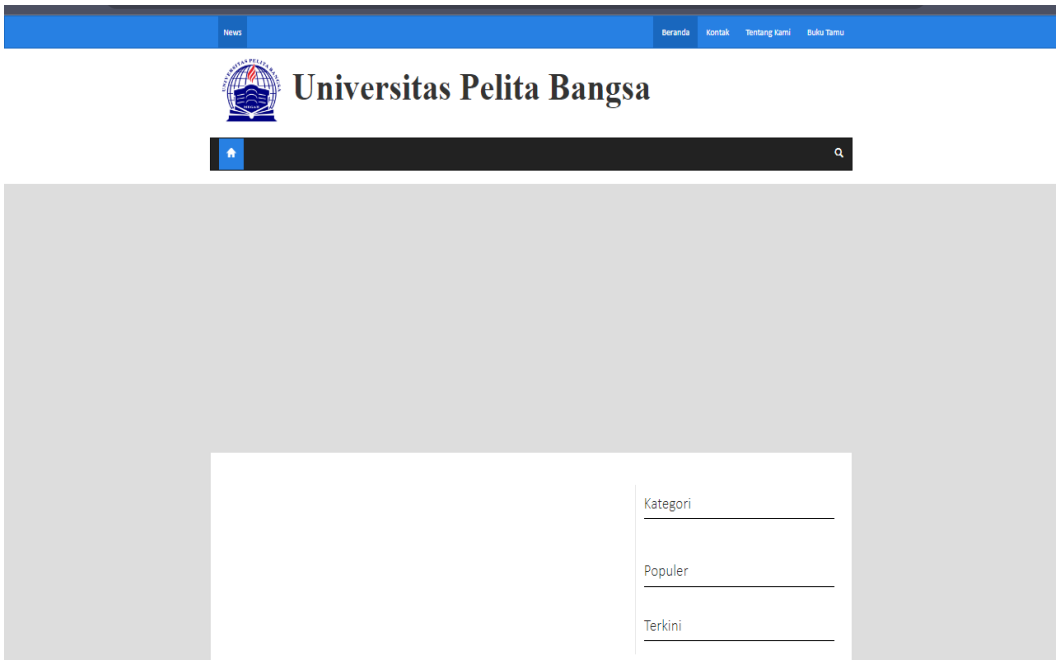
C. Data model



Data model (Alternatif)



D. User Interface



User Interface (Admin)