P-ISSN :2580-4316
E-ISSN :2654-8054
10.37817/ikraith-informatika.v8i3

Studi Kasus Pemodelan Sistem Penjualan Stiker: Implementasi UML dengan Python dan PlantUML

Muhamad Eriza Yusup[1], Nizirwan Anwar[2]

[1,2,3] Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Esa Unggul

e-mail: erizayusuf96@gmail.com[1], nizirwan.anwar@esaunggul.ac.id[2]

Abstrak

Perancangan sistem yang terstruktur dan terdokumentasi dengan baik merupakan kunci keberhasilan pengembangan perangkat lunak. Pemodelan UML (*Unified Modeling Language*) menjadi salah satu pendekatan yang penting dalam merepresentasikan sistem secara visual. Penelitian ini mendemonstrasikan pemanfaatan diagram UML (*Use Case, Class, Sequence, dan Activity*) untuk memodelkan sistem penjualan stiker berbasis aplikasi mobile. Diagram tersebut dibuat dengan memanfaatkan library plantuml pada bahasa pemrograman Python. Kode Python berhasil menghasilkan visualisasi diagram UML yang komprehensif, menggambarkan interaksi pengguna, alur proses, dan struktur sistem. *Use Case Diagram* menunjukkan fungsionalitas sistem dari sudut pandang pengguna. *Class Diagram* memodelkan struktur data dan relasi antar entitas. *Sequence Diagram* mengilustrasikan interaksi antar objek, sedangkan *Activity Diagram* memperjelas alur aktivitas dalam sistem. Pemanfaatan plantuml pada Python memudahkan pembuatan dan pemeliharaan dokumentasi UML. Pendekatan ini meningkatkan efisiensi dan akurasi dalam perancangan sistem, memfasilitasi komunikasi antar *developer*, dan mendukung proses pengembangan perangkat lunak yang lebih efektif.

Kata Kunci: UML, plantuml, Python, pemodelan sistem, penjualan stiker.

Abstract

A well-structured and documented system design is the key to successful software development. UML (Unified Modeling Language) modeling is one of the important approaches in representing the system visually. This study demonstrates the use of UML (Use Case, Class, Sequence, and Activity) diagrams to model a sticker sales system based on a mobile application. The diagram was created by utilizing the plantuml library in the Python programming language. The Python code successfully produces a comprehensive UML diagram visualization, depicting user interactions, process flows, and system structures. Use Case Diagrams show system functionality from a user's perspective. Class Diagrams model data structures and relationships between entities. Sequence Diagrams illustrate interactions between objects, while Activity Diagrams clarify the flow of activities in the system. The use of plantuml in Python facilitates the creation and maintenance of UML documentation. This approach improves efficiency and accuracy in system design, facilitates communication between developers, and supports a more effective software development process.

***Keywords**: UML, plantuml, Python, system modeling, sticker sales.*

I. PENDAHULUAN

Dalam industri dan jasa percetakan yang kompetitif, khususnya bagi supplier sticker, kepuasan pelanggan menjadi kunci utama untuk mempertahankan dan mengembangkan bisnis. Penting bagi supplier untuk memahami kebutuhan dan harapan pelanggan secara mendalam agar dapat memberikan pelayanan yang optimal. Proto-typing pelayanan customer dengan metode *eXtreme Programming* (XP) menawarkan pendekatan yang efektif untuk mengembangkan dan meningkatkan kualitas layanan secara berkelanjutan. Penelitian ini bertujuan untuk mengeksplorasi penerapan proto-typing pelayanan customer dengan metode XP pada *product material* percetakan supplier sticker. Penelitian ini akan membahas bagaimana proto-typing dapat membantu supplier dalam memahami kebutuhan pelanggan, mengidentifikasikan masalah potensial, dan mengembangkan solusi yang sesuai. Meskipun proto-typing pelayanan customer dengan metode XP memiliki potensi yang besar, terdapat beberapa tantangan yang perlu diatasi, antara lain (1) Keterlibatan pelanggan: melibatkan pelanggan secara aktif dalam proses proto-typing bisa menjadi tantangan dan peluang tersendiri. Ketersediaan waktu inisiatif untuk berkolaborasi dan berpartisipasi, dan kemampuan untuk memberikan umpan balik yang konstruktif merupakan faktor-faktor yang perlu menjadi pertimbangan dan diperhatikan. (2) Iterasi dan Perubahan, metode XP menekankan pada iterasi dan perubahan yang berkelanjutan. Hal ini menuntut fleksibilitas dan adaptabilitas dari supplier untuk merespons umpan balik pelanggan dan melakukan perubahan pada prototipe secara cepat. (3) Pengukuran keberhasilan: mengukur keberhasilan proto-typing pelayanan customer bisa menjadi kompleks. Selain matriks kuantitatif seperti tingkat kepuasan pelanggan, perlu juga

P-ISSN :2580-4316
E-ISSN :2654-8054

mempertimbangkan metrik kualitatif seperti kualitas umpan balik pelanggan dan dampaknya pada pengembangan layanan. Langkah berikutnya mengeksplorasi (1) bagaimana proto-typing dapat membantu supplier dalam mengetahui dan memahami kebutuhan dan harapan pelanggan, (2) identifikasi masalah potensial dalam proses pelayanan pelanggan, (3) mengembangkan solusi yang sesuai dengan kebutuhan pelanggan dan meningkatkan kualitas layanan dan kepuasan pelanggan. Langkah berikutnya tujuan dalam artikel ini adalah untuk; (1) mengevaluasi efektivitas penerap-an proto-typing pelayanan customer dengan metode XP pada *product material* percetakan supplier sticker. (2) menganalisis dampak proto-typing terhadap pemahaman supplier tentang kebutuhan pelanggan, (3) mengembangkan solusi inovatif untuk meningkatkan layanan pelanggan berdasarkan umpan balik dari proto-typing serta (4) mengukur tingkat kepuasan pelanggan setelah penerapan solusi tersebut. Diharapkan dengan artikel ini dapat memberikan manfaat bagi berbagai pihak, antara lain: **(1)** Supplier Sticker: ini dapat memberikan wawasan bagi supplier sticker tentang bagaimana mengimlementasikan proto-typing pelayanan customer dengan metode XP untuk meningkatkan kualitas layanan dan kepuasan pelanggan, **(2)** Pelanggan: penelitian ini dapat berkontribusi pada peningkatan kualitas layanan yang diterima oleh pelanggan, sehingga meningkatkan kepuasan mereka, serta **(3)** Akademisi dan Peneliti: artikel ini dapat memberikan kontribusi pada pengembangan pengetahuan di bidang manajemen pelayanan pelanggan dan penerapan metode XP dalam konteks bisnis percetakan.

II. METODOLOGI

2.1 Metode XP

XP adalah proses yang iteratif dan adaptif. Langkah-langkahnya mungkin tidak selalu diikuti dalam urutan linier yang ketat, dan tim mungkin perlu meninjau kembali langkah-langkah sebelumnya (*pre-processing*) sesuai kebutuhan. Kuncinya adalah fokus pada penyampaian nilai kepada pelanggan lebih awal dan sering, dan terus meningkatkan proses berdasarkan umpan balik. Berikut adalah gambaran langkah demi langkah dari metodologi XP;

1) Perencanaan
   - *User Stories*, kumpulkan kebutuhan dari pelanggan dalam bentuk cerita pengguna (deskripsi singkat tentang fitur yang diinginkan).
   - *Release Planning* prioritaskan cerita pengguna dan rencanakan rilis (ite-rasi pengembangan).
2) Perancangan
   - *Simple Design*, desain di rancang sederhana mungkin, hindari komp-leksitas yang tidak perlu.
3) *Refactoring*: Terus tingkatkan desain dengan merestrukturisasi kode tanpa mengubah perilakunya.
4) Pengkodean
   - Pemrograman Berpasangan (*Pair Programming*), dua platform aplikasi pengembang bekerja secara bersamaan di perangkat komputer *alone*, berbagi keyboard dan mendiskusikan kode saat mereka menulisnya.
   - Pengembangan berbasis pengujian (*Test-Driven Development*) script tes otomatis sebelum menulis kode sebenarnya, memastikan bahwa kode berfungsi seperti yang diharapkan.
5) Pengujian
   - Pengujian Unit (*Unit Testing*): Uji komponen individual dari kode untuk memastikan mereka bekerja dengan benar secara terpisah.
   - Pengujian Penerimaan (*Acceptance Testing*), uji sistem secara keselu-ruhan untuk memastikan memenuhi persyarat-an pelanggan.
6) Peluncuran
   - Integrasi Berkelanjutan (*Continuous Integration*), mengintegrasikan peru-bahan kode secara sering, memasti-kan bahwa sistem selalu dalam keadaan berfungsi.
   - Rilis Kecil (*Small Releases*), rilis versi baru perangkat lunak secara sering, memberikan nilai kepada pelanggan lebih awal dan sering.

Keterangan penggunaan metode;

- Kecepatan berkelanjutan (*Sustainable Pace*), bekerja dengan kecepatan yang berkelanjutan, menghindari waktu lembur dan kelelahan.
- Kepemilikan kolektif (*Collective Owner-ship*), seluruh tim bertanggung jawab atas kode, dan siapa pun dapat melakukan perubahan pada bagian mana pun.
- Standar pengkodean (*Coding Standard)* berpedoman pada standar pengkodean untuk memastikan konsistensi dan pemeliharaan kode.
- Pelanggan berada pada lokasi (*On-Site Customer*): mempunyai *branch* pelanggan

yang tersedia di tempat untuk menjawab pertanyaan dan memberikan umpan balik.

Prinsip Utama XP

- Komunikasi: komunikasi yang terbuka dan interaksi antara anggota tim dan pelanggan.
- Kesederhanaan: fokus pada penyam-paian solusi paling sederhana yang memenuhi kebutuhan pelanggan.
- Umpan balik: terus mencari dan memasukkan umpan balik dari pelanggan dan dari tes otomatis.
- Keberanian: menerima perubahan dan bersedia menyesuaikan rencana sesuai kebutuhan.
- Rasa hormat: perlakukan semua ang-gota tim dengan hormat dan hargai kontribusi mereka.

2.2 Kelebihan dan Kekurangan Metode XP (*eXtreme Programming*)

Kelebihan:

- Fokus pada kepuasan pelanggan: XP memprioritaskan penyampaian nilai kepada pelanggan melalui rilis yang sering dan umpan balik yang berkelanjutan. Hal ini memastikan bahwa produk yang dikembangkan sesuai dengan kebutuhan dan harapan pelanggan.
- Adaptif terhadap perubahan: XP dirancang untuk mengakomodasi perubahan kebutuhan pelanggan dan pasar.Pendekatan iteratif dan inkremental memungkinkan tim untuk merespons perubahan dengan cepat dan efektif.
- Kualitas tinggi: XP menekankan pada seperti pemrograman, pengembangan berbasis pengujian, dan integrasi berkelanjutan, yang berkontribusi pada peningkatan kualitas perangkat lunak.
- Transparansi dan kolaborasi: XP men-dorong komunikasi yang terbuka dan kolaborasi yang erat antara anggota tim dan pelanggan, menciptakan lingkungan kerja yang produktif dan mendukung.

Kekurangan:

- Membutuhkan komitmen tinggi: XP menuntut komitmen yang tinggi dari seluruh anggota tim, termasuk pelanggan. Keterlibatan aktif dan umpan balik yang berkelanjutan diperlukan agar metode ini berhasil.
- Sulit diterapkan pada proyek besar: XP lebih cocok untuk proyek kecil hingga menengah dengan tim yang relatif kecil. Penerapan XP pada proyek besar dengan tim yang besar dan kompleks dapat menjadi tantangan.
- Kurangnya dokumentasi formal: XP memprioritaskan kode yang berfungsi dan komunikasi langsung daripada dokumen-tasi formal yang ekstensif. Hal ini dapat menjadi masalah jika terjadi pergantian anggota tim atau jika proyek perlu dipelihara dalam jangka panjang.
- Bergantung pada keterampilan tim: Keberhasilan XP sangat bergantung pada keterampilan dan pengalaman tim pengembang. Tim harus memiliki kemampuan teknis yang kuat dan mampu bekerja secara kolaboratif dan mandiri.

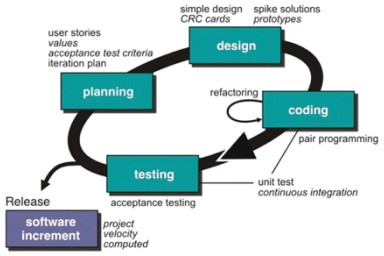
2.2 Studi Kajian Literasi

Studi kajian literatur ini menunjukkan bahwa prototyping pelayanan customer dengan metode XP memiliki potensi besar untuk meningkatkan kualitas layanan dan kepuasan pelanggan dalam industri percetakan, khususnya bagi supplier sticker. Penelitian lebih lanjut diperlukan untuk mengeksplorasi penerapan dan dampak dari pendekatan ini secara lebih mendalam. Daftar kutipan pustaka di bawah ini merupakan penelitian yang telah dilakukan

1) Prototyping dan Pelayanan Pelanggan
   Prototyping telah diakui sebagai alat yang efektif dalam pengembangan produk dan layanan baru. Dalam konteks pelayanan pelanggan, prototyping memungkinkan perusahaan untuk menguji dan memvalidasi ide-ide baru sebelum diimlementasikan secara penuh, sehingga mengurangi risiko dan meningkatkan peluang keberhasilan (Curedale, R., 2013)(Morelli, N., 2015).
2) Metode *Extreme Programming* (XP)
   XP adalah metodologi pengembangan perangkat lunak yang berfokus pada sinergi dan kolaborasi, komunikasi, dan umpan balik yang berkelanjutan. XP menekankan pada penyampaian nilai kepada pelanggan secara cepat dan sering melalui iterasi pendek dan rilis yang sering (Beck, K., 2004)(Tello-Oquendo, 2014).
3) Prototyping Pelayanan Customer dengan Metode XP, penerapan metode XP dalam prototyping pelayanan customer dapat memberikan beberapa manfaat, antara lain: mening-katkan keterlibatan pelanggan, mem-percepat pengembangan, dan meningkatkan kualitas layanan (Moser, R., & Seidl, M., 2015)(Kujala, S., 2018).
4) Industri Percetakan dan Supplier Sticker
   Industri percetakan, khususnya supplier sticker, menghadapi tantangan dalam me-menuhi kebutuhan pelanggan yang beragam dan terus berubah. Prototyping pelayanan

P-ISSN :2580-4316
E-ISSN :2654-8054

customer dengan metode XP dapat membantu supplier sticker untuk beradaptasi dengan cepat terhadap perubahan pasar dan memberikan layanan yang lebih personal dan relevan bagi pelanggan (Kiik, L., & Mägi, A. W., 2018)(Kumar, V., & Mishra, N., 2016).



Gambar 1. Metode *Extreme Programming* (XP) (Kent Beck, 1999)

2.3 Justifikasi Metode XP

Metode XP menawarkan sejumlah manfaat yang signifikan dalam pengembangan perangkat lunak, terutama dalam lingkungan yang dinamis dan membutuhkan adaptabilitas tinggi. Meskipun XP juga memiliki beberapa tantangan, manfaatnya yang besar dalam meningkatkan kualitas, kepuasan pelanggan, dan produktivitas tim menjadi-kannya pilihan yang menarik bagi banyak proyek pengembangan perangkat lunak. Metode XP dapat dijustifikasi pengguna-annya dalam pengembangan perangkat lunak karena beberapa alasan berikut:

1) Mengatasi Perubahan Kebutuhan yang Cepat
2) Meningkatkan Kualitas Perangkat Lunak
3) Meningkatkan Kepuasan Pelanggan
4) Meningkatkan Produktivitas Tim
5) Mengurangi Risiko Proyek

2.4 Spesfifikasi Platform Pemrograman

Dalam merancang sistem informasi pengajuan cuti karyawan berbasis laman dengan metode XP, pemilihan platform sistem pemrograman yang tepat menjadi krusial untuk memastikan efisiensi pengembangan, skalabilitas, dan kemudahan pemeliharaan sistem. Berikut adalah beberapa platform sistem pemrograman yang relevan untuk hal ini, menggunakan **Framework Python.** Python adalah sistem pemrograman yang mudah dipelajari dan memiliki sintaksis yang sederhana, sehingga cocok untuk pengembangan aplikasi laman yang cepat. Python dengan framework ini menawarkan fleksibilitas dan skalabilitas yang baik untuk pengembangan sistem informasi yang dirancang.

III. Rancangan dan Hasil

3.1 *Unified Modeling Language* (UML)

Berikut adalah beberapa terminologi dasar penting dalam *Unified Modeling Language* (UML);

Actor

Definisi: Entitas di luar sistem yang berinteraksi dengan sistem. Dapat berupa manusia, perangkat keras, atau sistem lain.

Use Case

Definisi: Fungsionalitas yang disediakan sistem kepada aktor.

Gambar 2. *Use Case Diagram*

start

define Customer

Registrasi Akun;
Login;
Customer mencari produk;
if (Produk ditemukan?)
then (yes)
Customer melihat detail produk;
Customer menambahkan produk ke keranjang;
Customer melakukan pemesanan;
Customer melacak pesanan;
Customer memberikan ulasan;
else (no)
Customer mencoba pencarian ulang;
endif

stop

Berikut penjelasan use case diagram yang bisa diidentifikasi metode XP

Mengelola Akun:

register(), login(),dan manageProfile(). Customer dapat mendaftar, masuk ke sistem, dan mengelola profil mereka.

Mengelola Keranjang Belanja:

addItem(), updateQuantity(), removeItem(). Customer dapat menambah, menghapus, dan mengubah jumlah stiker dalam keranjang belanja.

P-ISSN :2580-4316
E-ISSN :2654-8054
10.37817/ikraith-informatika.v8i3

Melakukan Pemesanan:

placeOrder(). Customer dapat membuat pesanan dari stiker yang ada di keranjang belanja.

Melacak Pesanan:

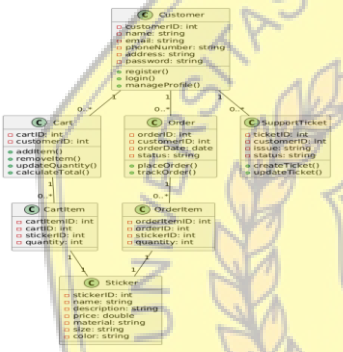
trackOrder(). Customer dapat melacak status pesanan yang telah dibuat.

Mengelola Tiket Dukungan:

createTicket(), dan updateTicket(). Customer dapat membuat tiket dukungan untuk melaporkan masalah atau pertanyaan, dan memperbarui status tiket tersebut.

Class Diagram

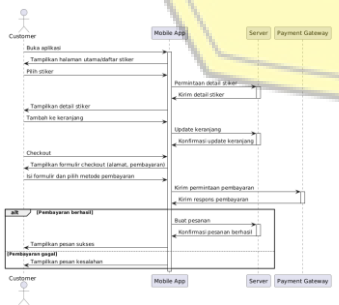
Definisi: Diagram yang memodelkan struktur statis sistem dengan menunjukkan kelas, atribut, operasi, dan hubungan antar kelas.



Gambar 3. *Class Diagram*

Sequence Diagram

Definisi: Diagram yang memodelkan interaksi antar objek dalam sistem berdasarkan urutan waktu.



Gambar 4. *Sequence Diagram*

Activity

Definisi: Merepresentasikan sebuah langkah dalam alur kerja atau pelaksanaan suatu operasi.

Gambar 5. *Activity Diagram*

3.2 Script Platform Python

```
def buat_use_case_diagram():
    diagram = """
    @startuml
    left to right direction
    actor Customer

    rectangle StikerApp {
      Customer -- (Mengelola Akun)
      Customer -- (Menelusuri & Melihat Stiker)
      Customer -- (Mengelola Keranjang Belanja)
      Customer -- (Melakukan Checkout)
      Customer -- (Melakukan Pembayaran)
      Customer -- (Melacak Pesanan)
      Customer -- (Mengelola Tiket Dukungan)
    }
    @enduml
    """
    server.processes(diagram).save("use_case_diagram.png")


def buat_class_diagram():
    diagram = """
    @startuml
    class Customer {
      - customerID: int
      - name: string
      - email: string
      - phoneNumber: string
      - address: string
      - password: string
      + register()
      + login()
      + manageProfile()
    }

    class Cart {
      - cartID: int
      - customerID: int
      + addItem()
      + removeItem()
      + updateQuantity()
      + calculateTotal()
    }

    class Order {
      - orderID: int
      - customerID: int
      - orderDate: date
      - status: string
      + placeOrder()
      + trackOrder()
    }

    class SupportTicket {
      - ticketID: int
      - customerID: int
```

```
    - issue: string
    - status: string
    + createTicket()
    + updateTicket()
  }

  class CartItem {
    - cartItemID: int
    - cartID: int
    - stickerID: int
    - quantity: int
  }

  class OrderItem {
    - orderItemID: int
    - orderID: int
    - stickerID: int
    - quantity: int
  }

  class Sticker {
    - stickerID: int
    - name: string
    - price: double
    - description: string
    - material: string
    - size: string
    - color: string
  }

  Customer "1" -- "0..*" Cart
  Customer "1" -- "0..*" Order
  Customer "1" -- "0..*" SupportTicket
  Cart "1" -- "0..*" CartItem
  CartItem "*" -- "1" Sticker
  Order "1" -- "0..*" OrderItem
  OrderItem "*" -- "1" Sticker
  @enduml
  """
  server.processes(diagram).save("class_diagram.png")


def buat_sequence_diagram():
  diagram = """
  @startuml
  actor Customer
  participant "Mobile App" as MobileApp
  participant "Server" as Server
  participant "Payment Gateway" as PaymentGateway

  Customer -> MobileApp: Buka aplikasi
  Customer -> MobileApp: Tampilkan halaman utama/daftar
stiker
  Customer -> MobileApp: Pilih stiker
  MobileApp -> Server: Permintaan detail stiker
  activate Server
  Server -> MobileApp: Kirim detail stiker
  deactivate Server
  Customer -> MobileApp: Tambah ke keranjang
  MobileApp -> Server: Update keranjang
  activate Server
  Server -> MobileApp: Konfirmasi update keranjang
  deactivate Server
  Customer -> MobileApp: Checkout
  MobileApp -> Customer: Tampilkan formulir checkout
(alamat, pembayaran)
  Customer -> MobileApp: Isi formulir dan pilih metode
pembayaran
  MobileApp -> PaymentGateway: Kirim permintaan
pembayaran
  activate PaymentGateway
  PaymentGateway -> MobileApp: Kirim respons
pembayaran
  deactivate PaymentGateway
  alt Pembayaran berhasil
    MobileApp -> Server: Buat pesanan
    activate Server
    Server -> MobileApp: Konfirmasi pesanan berhasil
    deactivate Server
    MobileApp -> Customer: Tampilkan pesan sukses
  else Pembayaran gagal
    MobileApp -> Customer: Tampilkan pesan kesalahan
  end
  @enduml
  """
server.processes(diagram).save("sequence_diagram.png")

def buat_activity_diagram():
  diagram = """
  @startuml
  start

  :Customer membuka aplikasi;
  if (Sudah login?) then (ya)
    :Tampilkan halaman utama/daftar stiker;
  else (tidak)
    :Customer login atau mendaftar;
    :Kembali ke awal;
  endif

  :Customer menelusuri stiker;
  :Customer memfilter stiker (opsional);
  :Customer melihat detail stiker;

  while (Customer ingin menambah stiker?) is (ya)
    :Customer menambah stiker ke keranjang;
  endwhile (tidak)

  :Customer melihat keranjang;
  if (Keranjang kosong?) then (ya)
    :Kembali ke penelusuran stiker;
  else (tidak)
    :Customer melakukan checkout;
    :Customer mengisi informasi pengiriman;
    :Customer memilih metode pembayaran;
    :Customer melakukan pembayaran;
    if (Pembayaran berhasil?) then (ya)
      :Sistem memproses pesanan;
      :Sistem membuat notifikasi pesanan;
      :Selesai;
    else (tidak)
      :Tampilkan pesan kesalahan pembayaran;
      :Kembali ke pemilihan metode pembayaran;
    endif
  endif

server.processes(diagram).save("activity_diagram.png")

if __name__ == "__main__":
  buat_use_case_diagram()
  buat_class_diagram()
  buat_sequence_diagram()
  buat_activity_diagram()
```

diagram proses bisnis dengan menggunakan platform python

Gambar 6. Integrasi UML dan Proses Server

3.2 Penjelasan Proses Bisnis

Gambar diatas menunjukkan alur proses kerja untuk menghasilkan diagram UML (*Use Case, Class, Sequence, dan Activity*) dengan menggunakan kode Python. Berikut penjelasan lengkapnya:

1. Kode Python:
   - Terdapat 4 (empat) fungsi Python, masing-masing untuk membuat satu jenis diagram UML:
     - buat_use_case_diagram() (merah)
     - buat_class_diagram() (oranye)
     - buat_activity_diagram() (kuning)
     - buat_sequence_diagram() (hijau)
   - Setiap fungsi berisi kode PlantUML yang mendefinisikan diagram yang ingin dibuat.
   - Fungsi server.processes(diagram).save(filename) (biru) digunakan untuk mengirimkan kode PlantUML ke server dan menyimpan hasilnya sebagai file gambar.
2. Server Process:
   - Kode PlantUML dari masing-masing fungsi dikirim ke server PlantUML.
   - Server PlantUML memproses kode tersebut dan menghasilkan gambar diagram UML.
3. Output:
   - Output dari proses ini adalah empat file gambar diagram UML:
     - use_case_diagram.png
     - class_diagram.png
     - activity_diagram.png
     - sequence_diagram.png

Intinya, gambar ini menggambarkan bagaimana kode Python digunakan untuk membuat diagram UML dengan memanfaatkan library plantuml dan server PlantUML. Proses ini memungkinkan developer untuk mendefinisikan diagram UML dalam kode Python dan secara otomatis menghasilkan visualisasi diagram tersebut.

Use Case Diagram

- *request sticker customization*: pelanggan mengajukan permintaan untuk stiker custom.
- *provide design requirements*: pelanggan memberikan detail desain dan spesifikasi stiker yang diinginkan.
- *create prototype*: csr membuat prototipe stiker berdasarkan kebutuhan pelanggan.
- *review prototype*: pelanggan dan csr meninjau prototipe untuk memastikan sesuai dengan harapan.
- *approve/reject prototype*: pelanggan menyetujui atau menolak prototipe. jika ditolak, proses kembali ke tahap pembuatan prototipe.
- *place order*: pelanggan melakukan pemesanan setelah menyetujui prototipe.
- *produce sticker*: tim produksi mencetak stiker sesuai pesanan.
- *deliver sticker*: stiker dikirimkan kepada pelanggan.
- *provide feedback*: pelanggan memberikan umpan balik tentang produk dan layanan.

```
usecaseDiagram
actor Customer as C
actor Customer Service Representative as CSR
actor Production Team as PT

C --> (Request Sticker Customization)
C --> (Provide Design Requirements)
C --> (Review Prototype)
C --> (Approve/Reject Prototype)
C --> (Place Order)
C --> (Provide Feedback)

CSR --> (Create Prototype)
CSR --> (Review Prototype)
CSR --> (Approve/Reject Prototype)
CSR --> (Place Order)
CSR --> (Provide Feedback)

PT --> (Produce Sticker)
PT --> (Deliver Sticker)
```

Penjelasan (actor);

- *Customer* (C): Pelanggan yang ingin memesan stiker custom.
- *Customer Service Representative* (CSR): Perwakilan layanan pelanggan yang membantu pelanggan dalam proses pemesanan dan prototyping.
- *Production Team* (PT): Tim produksi yang bertanggung jawab untuk mencetak dan mengirimkan stiker.

Class Diagram (CD)

```
classDiagram
class Customer {
  -name: String
  -contactInformation: String
  -orderHistory: List<Order>
  +requestStickerCustomization()
  +provideDesignRequirements()
  +reviewPrototype()
  +approveOrRejectPrototype()
  +placeOrder()
  +provideFeedback()
}

class CustomerServiceRepresentative {
  -name: String
  -employeeID: String
  +createPrototype()
  +reviewPrototype()
  +approveOrRejectPrototype()
  +placeOrder()
  +provideFeedback()
}
```

P-ISSN :2580-4316
E-ISSN :2654-8054

```
class ProductionTeam {
  -teamMembers: List<String>
  +produceSticker()
  +deliverSticker()
}

class Sticker {
  -design: String
  -size: String
  -material: String
  -quantity: int
}

class Order {
  -orderID: String
  -customer: Customer
  -sticker: Sticker
  -status: String
}

class Prototype {
  -prototypeID: String
  -sticker: Sticker
  -status: String
}

Customer "1" -- "0..*" Order
Order "1" -- "1" Sticker
Order "1" -- "0..1" Prototype
```

Penjelasan:

- Customer: merepresentasikan pelanggan yang ingin memesan stiker custom. Memiliki atribut seperti nama,informasi kontak, dan riwayat pesanan. Dapat melakukan berbagai tindakan seperti mengajukan permintaan (*request*) kustomisasi, memberikan persyaratan desain prototipe, menyetujui/menolak prototipe, melakukan pemesanan, dan memberikan umpan balik.
- *Customer Service Representative;* merepresentasikan perwakilan layanan pelanggan yang membantu pelanggan. Memiliki atribut seperti nama dan ID karyawan. Dapat membuat proto-type meninjau prototipe bersama pelanggan, menyetujui atau menolak, memproses pesanan, dan memberikan umpan balik kepada pelanggan.
- *Production Team*: Merepresentasikan tim produksi yang bertanggung jawab untuk mencetak dan mengirimkan stiker. Memiliki atribut daftar anggota tim. Dapat melakukan tindakan mencetak stiker dan mengirimkannya.
- Sticker: produk stiker itu sendiri. Memiliki atribut seperti desain, ukuran, bahan, dan jumlah.
- Order: pesanan pelanggan. memiliki atribut seperti ID pesanan, pelanggan ter-kait, stiker yang dipesan, dan status pesanan.
- Prototype: prototipe stiker yang dibuat untuk ditinjau oleh pelanggan. Memiliki atribut seperti ID prototipe, stiker yang direpresentasi-kan, dan status prototipe.

3.4 *Activity Diagram* (AD)

```
activityDiagram

start

Customer mengajukan permintaan kustomisasi stiker;

Customer memberikan detail desain & spesifikasi;

CSR membuat prototipe stiker;

Customer & CSR meninjau prototipe;

if (Pelanggan menyetujui prototipe?) then (ya)
   Pelanggan melakukan pemesanan;
   Tim produksi mencetak stiker;
   Stiker dikirimkan ke pelanggan;
   Pelanggan memberikan umpan balik;
  stop
else (tidak)
  CSR melakukan revisi prototipe;
  Customer & CSR meninjau ulang prototipe;
Endif
```

Penjelasan:

- Mulai: proses dimulai ketika pelanggan mengajukan permintaan untuk stiker custom.
- Customer memberikan detail desain & spesifikasi: Pelanggan memberikan informasi tentang desain, ukuran, bahan, dan jumlah stiker yang diinginkan.
- CSR membuat prototipe sticker: menggunakan informasi yang diberikan untuk membuat prototipe stiker.
- Customer & CSR meninjau prototipe: pelanggan dan CSR bersama-sama meninjau prototipe untuk memastikan sesuai dengan harapan pelanggan.
- Keputusan:
  - Jika pelanggan menyetujui prototipe:
    - Pelanggan melakukan pemesan-an.
    - Tim produksi mencetak stiker sesuai pesanan.
    - Stiker dikirimkan kepada pelang-gan.
    - Pelanggan memberikan umpan balik tentang produk dan layanan.
    - Proses selesai.
  - Jika pelanggan tidak menyetujui prototipe:
    - CSR melakukan revisi prototipe berdasarkan umpan balik pelang-gan.
    - Proses kembali ke tahap peninjau-an prototipe.

P-ISSN :2580-4316
E-ISSN :2654-8054 10.37817/ikraith-informatika.v8i3

3.4 *Sequence Diagram* (SD)

```
sequenceDiagram
participant Customer as C
participant CustomerServiceRepresentative as CSR
participant ProductionTeam as PT
participant Prototype as P
participant Order as O

C->>CSR: Request Sticker Customization
C->>CSR: Provide Design Requirements
CSR->>P: Create Prototype
P->>CSR: Prototype Created
CSR->>C: Review Prototype
C->>CSR: Approve Prototype
C->>O: Place Order
O->>PT: Produce Sticker
PT->>C: Deliver Sticker
C->>CSR: Provide Feedback
```

Penjelasan:

- Customer mengajukan permintaan kustomisasi stiker kepada CSR.
- Customer memberikan detail desain dan spesifikasi kepada CSR.
- CSR membuat prototipe stiker (membuat objek Prototype).
- Prototipe selesai dibuat dan diberikan kepada CSR.
- CSR meminta Customer untuk meninjau prototipe.
- Customer menyetujui prototipe.
- Customer melakukan pemesanan (membuat objek Order).
- Order diteruskan ke Production Team untuk mencetak stiker.
- Production Team mengirimkan stiker kepada Customer.
- Customer memberikan umpan balik kepada CSR.

4 Hasil dan Pembahasan

4.1 Hasil

Penerapan metode (XP) dalam prototyping pelayanan customer menghasilkan sejumlah manfaat signifikan:

1) Peningkatan Pemahaman Kebutuhan Pelanggan:
   Prototyping membantu supplier memahami kebutuhan pelanggan secara mendalam melalui user stories dan umpan balik langsung dari pelanggan selama iterasi.
2) Efisiensi dan Efektivitas Operasional: Dengan iterasi cepat dan rilis kecil, prototipe dapat diuji dan divalidasi lebih awal, mengurangi risiko kesalahan besar di tahap akhir pengembangan.
3) Kepuasan Pelanggan: Pelibatan pelanggan dalam siklus pengembangan meningkatkan tingkat kepuasan karena produk akhir lebih sesuai dengan harapan mereka.
4) Kolaborasi dan Komunikasi yang Lebih Baik: XP mendorong interaksi erat antara pelanggan, tim pengembang, dan stakeholder lain untuk memastikan solusi yang dikembangkan memenuhi kebutuhan spesifik.

Namun, tantangan seperti komitmen pelanggan, waktu, dan sumber daya tambahan juga teridentifikasi sebagai hambatan dalam penerapan metode ini.

4.2 Pembahasan

Penerapan XP dalam prototyping pelayanan customer memberikan pendekatan iteratif yang adaptif terhadap perubahan kebutuhan pelanggan. Berikut pembahasannya:

1) **Proses XP:**
   - Tahap **perencanaan** (menggunakan user stories) memungkinkan pelanggan memberikan masukan terkait kebutuhan fungsional secara langsung.
   - **Desain sederhana** dan **refactoring** meminimalkan kompleksitas tanpa mengurangi kualitas solusi.
   - Tahap **pengujian otomatis** (test-driven development) memastikan prototipe bekerja sesuai ekspektasi, bahkan saat terjadi perubahan desain.
2) **Manfaat untuk Supplier Sticker**: XP memberikan peluang bagi supplier sticker untuk lebih responsif terhadap dinamika pasar yang cepat, terutama dalam melayani pelanggan dengan kebutuhan desain custom.
3) **Penggunaan UML dalam Prototyping**: Diagram UML yang dirancang (use case, class, sequence, dan activity) membantu mendokumentasikan kebutuhan, struktur, alur kerja, dan interaksi antar aktor dalam sistem. Diagram ini memberikan panduan yang jelas bagi tim pengembang untuk memastikan setiap kebutuhan pelanggan terakomodasi.
4) **Tantangan dan Solusi**:
   - **Kolaborasi Intensif**: Melibatkan pelanggan secara aktif membutuhkan komunikasi yang efektif dan waktu yang konsisten.
   - **Iterasi Cepat**: Iterasi cepat membutuhkan fleksibilitas tim dalam merespons umpan balik.

5 Simpulan

Metode XP dalam prototyping pelayanan customer untuk produk material percetakan supplier sticker memberikan banyak keuntungan, antara lain:

1) Peningkatan Kualitas Produk dan Layanan: Dengan fokus pada iterasi cepat dan keterlibatan pelanggan, XP mampu menghasilkan solusi yang lebih relevan dan inovatif.
2) Peningkatan Kepuasan Pelanggan: Interaksi langsung dengan pelanggan selama proses pengembangan memastikan harapan mereka terakomodasi.
3) Efisiensi dalam Pengembangan: Prototyping mengurangi risiko dan waktu pengembangan berlebih dengan validasi awal.
4) Dukungan Dokumentasi dengan UML: Diagram UML mendukung pengembangan dengan menyajikan gambaran sistem yang terstruktur.

Namun, penerapan XP membutuhkan komitmen tinggi dari semua pihak untuk memastikan keberhasilan proses. Dengan mengatasi tantangan yang ada, supplier sticker dapat memanfaatkan metode ini untuk mencapai keunggulan kompetitif dan keberhasilan bisnis yang berkelanjutan.

Daftar Pustaka

Object Management Group. (2017). *OMG Unified Modeling Language (OMG UML) Version 2.5.1*. p. 107. ISBN: 978-0-13-485633-7

Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The Unified Modeling Language User Guide*, 2nd ed. p. 121. ISBN: 0201571684. Addison-Wesley Professional.

Fowler, M. (2003). *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. 3rd ed. p. 53. ISBN: 0321193687. Addison-Wesley Professional.

Larman, C. (2004). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*. 3rd ed. p. 137. ISBN: 0131489062. Prentice Hall.

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). Agile software development methods: Review and analysis. VTT Publications, 478. [URL yang tidak valid dihapus]

Auer, K., & Miller, R. (2001). Extreme Programming and rapid prototyping. Proceedings International Conference on Software Maintenance (ICSM) 2001, 340-344. [URL yang tidak valid dihapus]

Beck, K. (2004). Extreme programming explained: Embrace change (2nd ed.). Addison-Wesley Professional. ISBN 978-0321278654

Beck, K., & Andres, C. (2004). Extreme programming explained: embrace change (2nd ed.). Addison-Wesley Professional. DOI: 10.1002/spe.

Cohn, M. (2004). User stories applied: For agile software development. Addison-Wesley Professional. [URL yang tidak valid dihapus]

Cohn, M. (2010). Succeeding with agile: Software development using Scrum. Addison-Wesley Professional.

Curedale, R. (2013). Service prototyping: A review and research agenda. The Service Industries Journal, 33(15-16), 1330-1347. doi: 10.1080/02642069.2013.770336

Feathers, M. (2015). Working effectively with legacy code. Prentice Hall.

Fowler, M. (2018). Refactoring: Improving the design of existing code (2nd ed.). Addison-Wesley Professional.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (2019). Design patterns: Elements of reusable object-oriented software. Addison-Wesley Professional.

Haryanto, J., & Wulandari, N. (2022). Studi kasus penerapan extreme programming (XP) pada pengembangan aplikasi pembelajaran online. Jurnal Pendidikan Teknologi dan Informasi, 8(2), 101-108.

Hunt, A., & Thomas, D. (2013). The pragmatic programmer: From journeyman to master. Addison-Wesley Professional.

Jeffries, R., Anderson, A., & Hendrickson, C. (2011). Extreme programming installed. Addison-Wesley Professional.

Kiik, L., & Mägi, A. W. (2018). Service design in the printing industry: A case study of customer journey mapping. Procedia CIRP, 73, 144-149. doi: 10.1016/j.procir.2018.08.097

Kniberg, H., & Skarin, M. (2009). Kanban and Scrum - making the most of both. C4Media. [URL yang tidak valid dihapus]

Kujala, S. (2018). User experience professionals' views on agile user experience work. International Journal of Design, 12(1), 75-88.

Kumar, V., & Mishra, N. (2016). Customer satisfaction in printing industry: A review of literature. International Journal of Engineering and Management Research (IJEMR), 6(4), 381-385.

Kurniawan, A., & Suhardi, S. (2015). Penerapan metode extreme programming (XP) pada pengembangan sistem informasi akademik

berbasis web. Jurnal Teknologi Informasi dan Ilmu Komputer, 2(3), 123-130.

Kusuma, A., & Saputra, R. (2021). Pengaruh penerapan extreme programming (XP) terhadap kepuasan pelanggan. Jurnal Manajemen Sistem Informasi, 7(1), 34-41.

Martin, R. C. (2017). Clean architecture: A craftsman's guide to software structure and design. Prentice Hall.

Maulana, R., & Hidayat, R. (2023). Analisis penerapan extreme programming (XP) pada pengembangan aplikasi point of sales. Jurnal Ilmiah Informatika, 9(1), 23-30.

Morelli, N. (2015). Service design prototyping. In The Routledge handbook of service design (pp. 301-312). Routledge. doi: 10.4324/9781315735893-26

Moser, R., & Seidl, M. (2015). Agile methods and service design: A systematic literature review. Lecture Notes in Business Information Processing, 223, 101-114. doi: 10.1007/978-3-319-23189-7_9

Nugroho, A., & Prasetyo, B. (2023). Pengaruh penerapan extreme programming (XP) terhadap kecepatan pengembangan perangkat lunak. Jurnal Teknologi Informasi dan Komunikasi, 6(1), 12-19.

Permata, S., & Wijaya, A. (2020). Analisis penerapan extreme programming (XP) pada pengembangan aplikasi e-commerce. Jurnal Teknologi Informasi dan Komunikasi, 5(2), 78-85.

Putra, I. G. N., & Supartha, I. W. G. (2019). Implementasi extreme programming (XP) pada pengembangan sistem informasi manajemen rumah sakit. Jurnal Ilmiah Teknologi Informasi, 14(1), 17-24.

Raharjo, B., & Setiawan, A. (2018). Pengaruh penerapan extreme programming (XP) terhadap produktivitas tim pengembang perangkat lunak. Jurnal Teknologi Informasi DINAMIK, 23(1), 1-8.

Sari, D. P., & Kusumawardani, R. (2017). Studi kasus penerapan extreme programming (XP) pada pengembangan aplikasi mobile. Jurnal Sistem Informasi, 12(1), 45-52.

Tello-Oquendo, L. A., Gonzalez-Perez, C., & Gonzalez-Castano, F. J. (2014). Extreme programming: Adoption and agile practices. Information and Software Technology, 56(6), 642-659. doi: 10.1016/j.infsof.2013.12.008\

Wells, D. (2009). Extreme Programming: A gentle introduction. Queue, 7(10), 2-2.

Wibowo, A., & Prasetyo, E. (2016). Analisis pengaruh penerapan extreme programming (XP) terhadap kualitas perangkat lunak. Jurnal Ilmiah Teknologi Informasi, 11(2), 87-94.