Motela

เสนอ
ดร. ปริญญา เอกปริญญา

จัดทำโดย

63010356 ดุษฎี ส่งเกษรชาติ
63010377 ทัตเทพ ชำนาญกิจ
63010382 ทิวัตถ์ โพธิ์ศรี
63010445 ธรณินท์ พงษ์สถิตย์พร
63010522 นาวีวัฒน์ พฤกพัฒนาชัย
63010548 บูรพา ทิมแดง
63010630 พชรพล จารุวนาวัฒน์
63010650 พฤกษ์ อภิรมรัตน์

รายงานนี้เป็นส่วนหนึ่งของวิชา 01076024 Software architecture and design คณะวิศวกรรมคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ที่มาและความสำคัญ

ปัญหาการเดินทางและการท่องเที่ยวแบบค้างคืนหรือระยะไกลนั้น สิ่งที่เราจำเป็นต้องการหาคือสถานที่พักผ่อนที่ตรงตามต้องการ ดี ราคาถูก และปลอดภัย ซึ่งเป็นเรื่องที่ยากที่จะทำการจองได้สำเร็จทั้งเรื่องการหาสถานที่ที่จะต้องค้นหาทีละเว็ปไซต์ เรื่องการโทรไปจองซึ่งเราไม่สามาราทราบได้เลยว่าเต็มถ้าไม่ติดต่อไปก่อน เรื่องการชำระเงินเป็นต้น เนื่องจากพวกเราเล็งเห็นถึงปัญหานี้เราจึงจะทำเว็บแอพพลิเคชั่นที่ขั้นตอนในการจองที่ง่าย มิลิสต์โรงแรมให้เลือกมากมาย ให้ผู้ใช้ได้เลือกโรงแรมที่ดีที่สุดสำหรับผู้ใช้

กลุ่มของพวกเราอยากที่จะพัฒนา Website application ที่ช่วยผู้ใช้สามารถหาโรงแรมที่ปลอดภัยได้ง่ายขึ้น และทางโรงแรมสามารถโปรโมทโรงแรมได้อีกด้วย โดย Website application ของเราจะประกอบไปด้วยระบบหลักๆดังนี้

1. ระบบการจัดการด้านการจองห้องพัก(Reservation management)
2. ระบบการจัดการด้านสถานะของห้องพัก
3. ระบบการสร้าง Profile ของโรงแรม

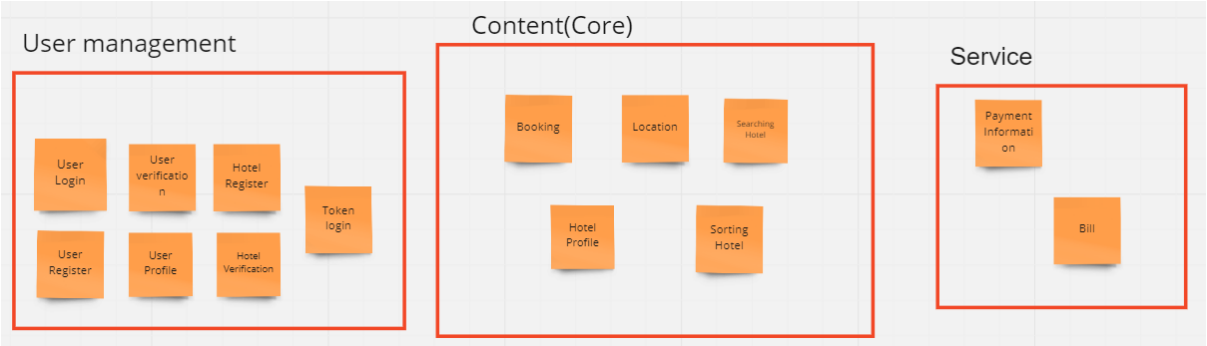
สมมุติฐาน

1. สามารถจองห้องพักได้
2. มีการคำณวณราคาของห้องพักได้
3. มีการ add โรงแรมขึ้น Website application ได้

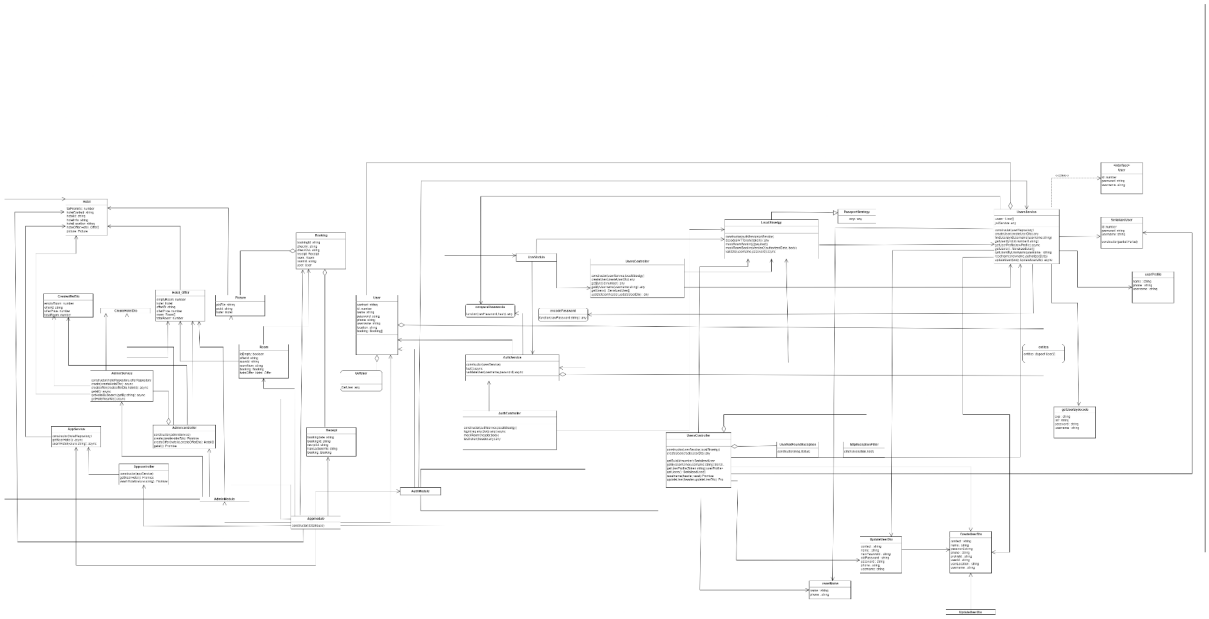
Boundaries

1. สิ่งที่ทำ
    - ระบบสมาชิกของผู้ใช้จองโรงแรมหรือเจ้าของโรงแรมอยากจะมาเพิ่มโรงแรมตัวเองที่ไม่ได้อยู่บน Website
    - ระบบสามารถค้นหาและจองห้องพัก โดยหาตามวันที่และประเภทห้องที่รองรับ
    - สามารถบันทึก Bill ย้อนหลังจากการจองโรมแรม
2. สิ่งที่ไม่ทำ
    - การใช้ code สำหรับส่วนลด
    - การจ่ายเงิน online

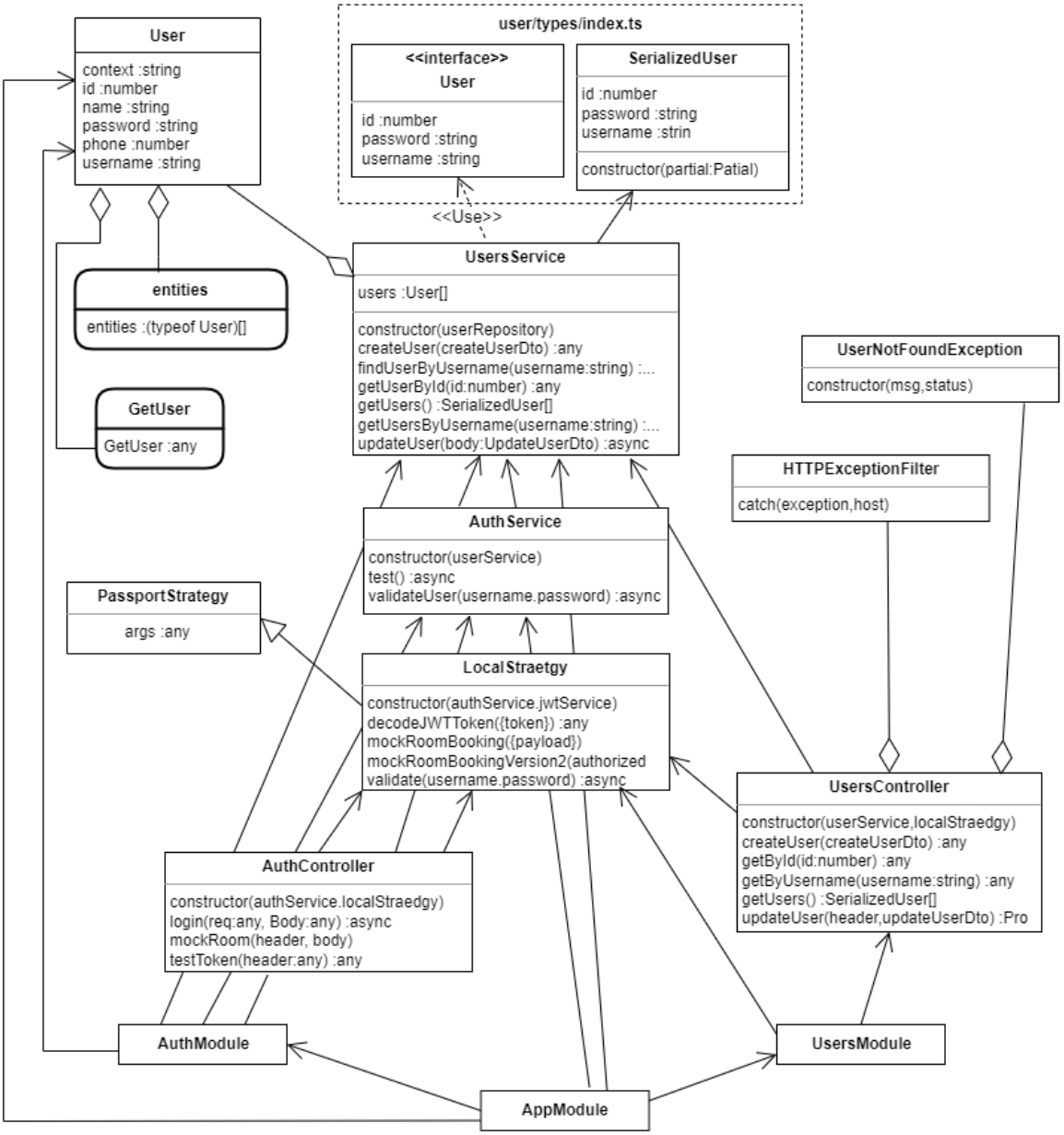
Bounded contexts



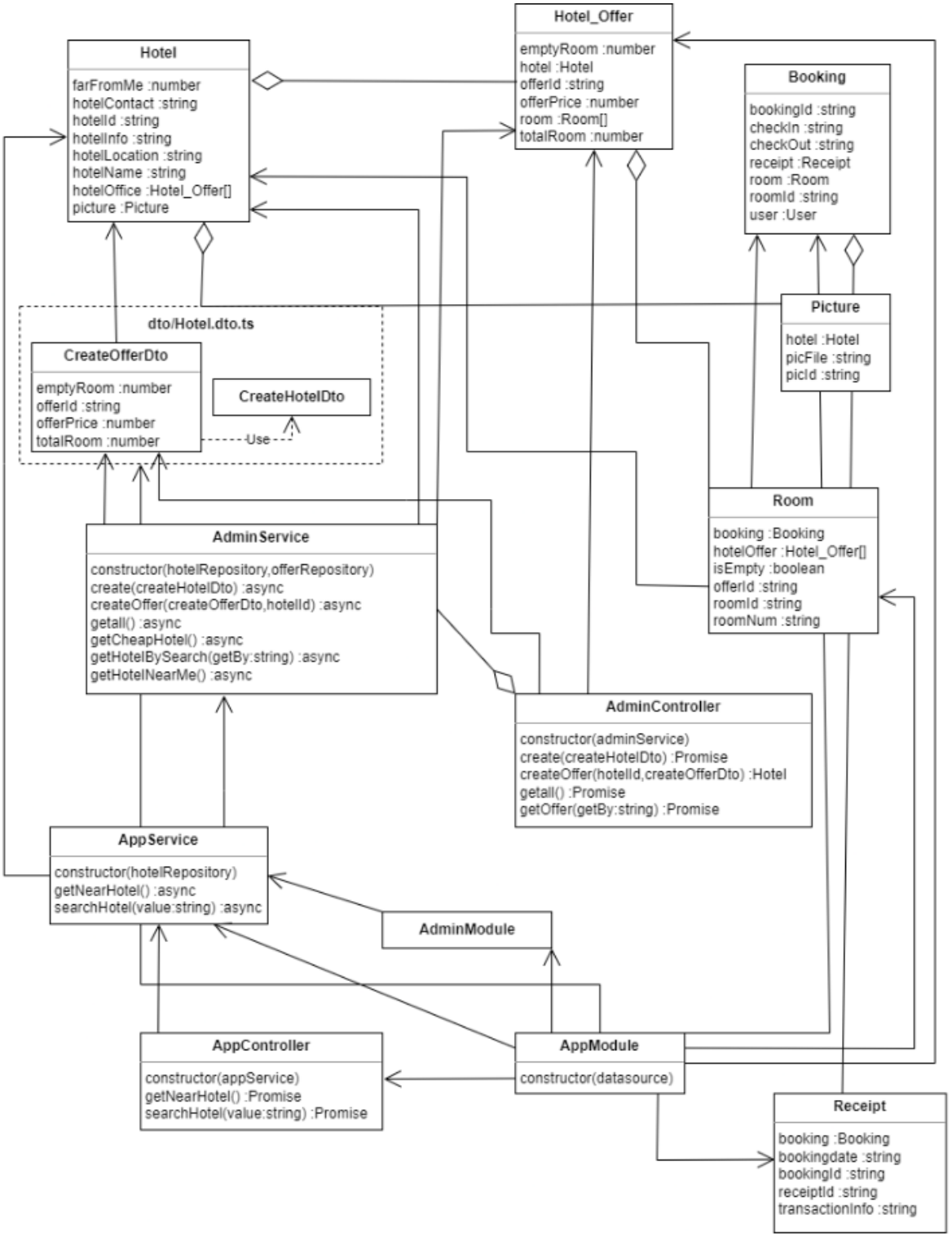
UML ของสถาปัตยกรรม



UML ของ Hotel



UML ของ User


Hotel_Offer
emptyRoom :number
hotel :Hotel
offerId :string
offerPrice :number
room :Room[]
totalRoom :number
Hotel
farFromMe :number
hotelContact :string
hotelId :string
hotelInfo :string
hotelLocation :string
hotelName :string
hotelOffice :Hotel_Offer[]
picture :Picture
Booking
bookingId :string
checkIn :string
checkOut :string
receipt :Receipt
room :Room
roomId :string
user :User
Picture
hotel :Hotel
picFile :string
picId :string
dto/Hotel.dto.ts
CreateOfferDto
emptyRoom :number
offerId :string
offerPrice :number
totalRoom :number
CreateHotelDto
Use
Room
booking :Booking
hotelOffer :Hotel_Offer[]
isEmpty :boolean
offerId :string
roomId :string
roomNum :string
AdminService
constructor(hotelRepository,offerRepository)
create(createHotelDto) :async
createOffer(createOfferDto,hotelId) :async
getall() :async
getCheapHotel() :async
getHotelBySearch(getBy:string) :async
getHotelNearMe() :async
AdminController
constructor(adminService)
create(createHotelDto) :Promise
createOffer(hotelId,createOfferDto) :Hotel
getall() :Promise
getOffer(getBy:string) :Promise
AppService
constructor(hotelRepository)
getNearHotel() :async
searchHotel(value:string) :async
AdminModule
AppController
constructor(appService)
getNearHotel() :Promise
searchHotel(value:string) :Promise
AppModule
constructor(datasource)
Receipt
booking :Booking
bookingdate :string
bookingId :string
receiptId :string
transactionInfo :string

Design Pattern ของ MOTELLA

1.Prototype

ปัญหาที่พบคือ ไม่สามารถประกาศ entity เป็น global ได้

จากนั้นจึงแก้ปัญหาด้วย วิธี prototype

Original entity ถูกสร้างขึ้นและมีการสร้าง entity ขึ้นมาใหม่โดยอ้างอิงจาก original entity และ entity ที่ถูกสร้างขึ้นมาใหม่สามารถถูกเรียกใช้ใน function อื่นได้

```
export class UsersService {
    constructor(@InjectRepository(UserEntity)private readonly userRepository: Repository<UserEntity>){

    }
    private users:User[]=[];
    getUsers(){
        return this.users.map((user) => new SerializedUser(user));
    }
```

2.Composite

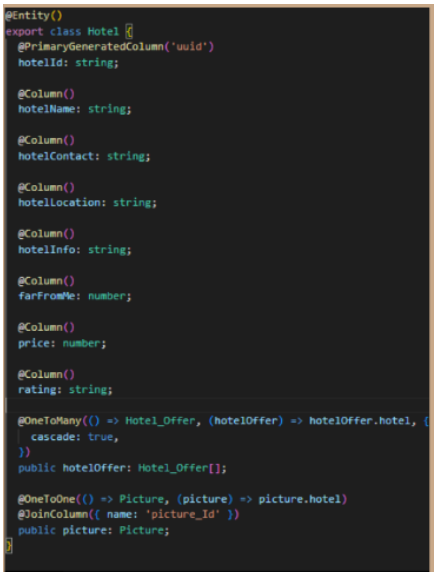
ปัญหาที่พบคือ ไม่สามารถรู้ได้ว่าภายใน hotel มี offter ใดๆบ้าง จากนั้นจึงแก้ปัญหาด้วย วิธี composite Composite ทำให้โยง relation ไปทาง object ที่เล็กกว่าโดยใช้ query

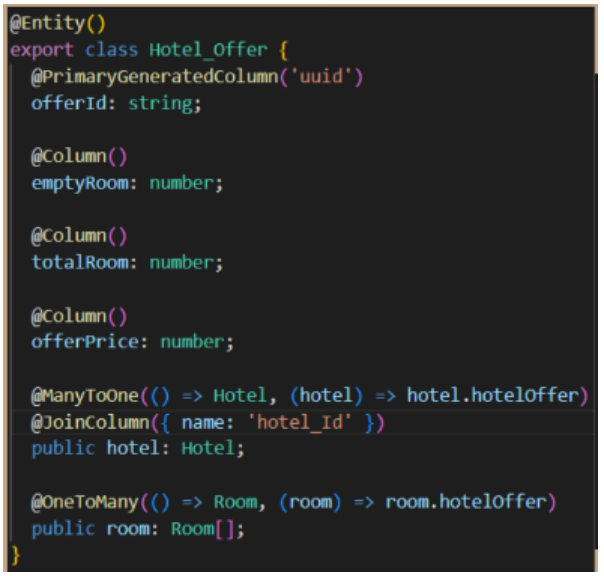
```
async createOffer(createOfferDto: CreateOfferDto, hotelId: string) {
  const offer = new Hotel_Offer();
  const hotel = this.hotelRepository.findOneBy({ hotelId });
  offer.offerId = createOfferDto.offerId;
  offer.emptyRoom = createOfferDto.emptyRoom;
  offer.totalRoom = createOfferDto.totalRoom;
  offer.offerPrice = createOfferDto.offerPrice;
  offer.hotel = await hotel;
  this.offerRepository.save(offer);
  return offer;
}

async getall() {
  const hotel = await this.hotelRepository
    .createQueryBuilder('hotel')
    .leftJoinAndSelect('hotel.hotelOffer', 'hotelOffer')
    .getMany();
  return hotel;
}
```

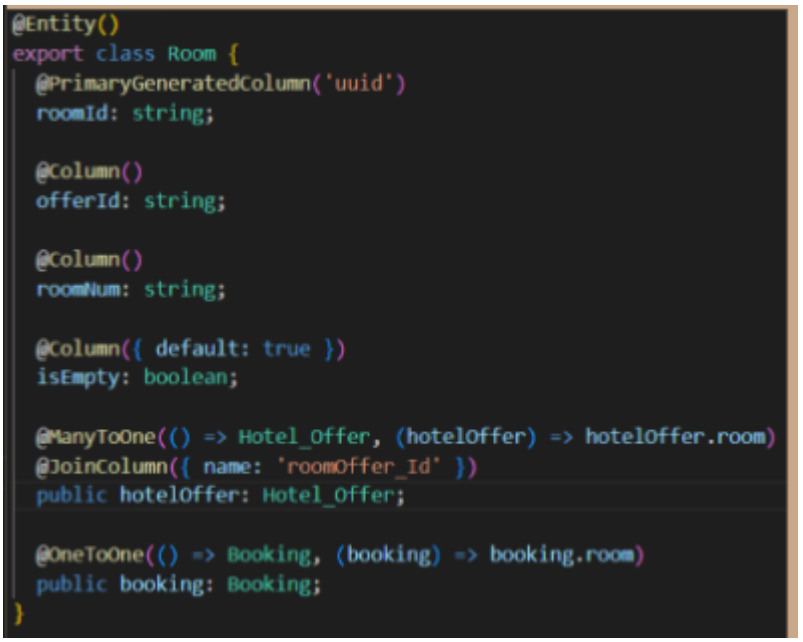
3.Mediator

ปัญหาที่พบคือ หา object ตัวรองไม่เจอจากนั้นจึงแก้ปัญหาด้วย วิธี Mediator

Mediator ทำให้สามารถเชื่อมต่อ object ตัวกลางกับตัวรองได้

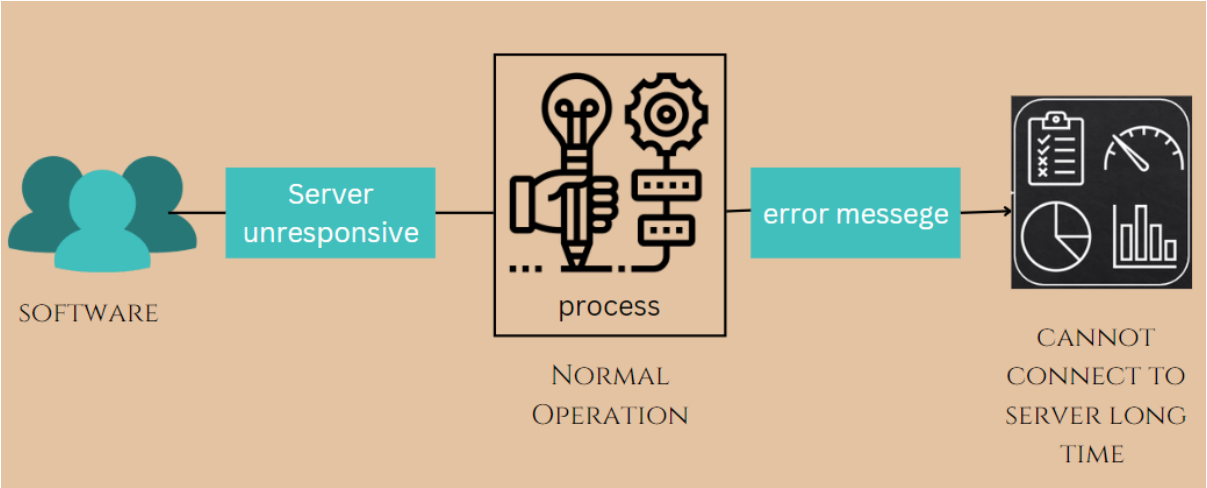




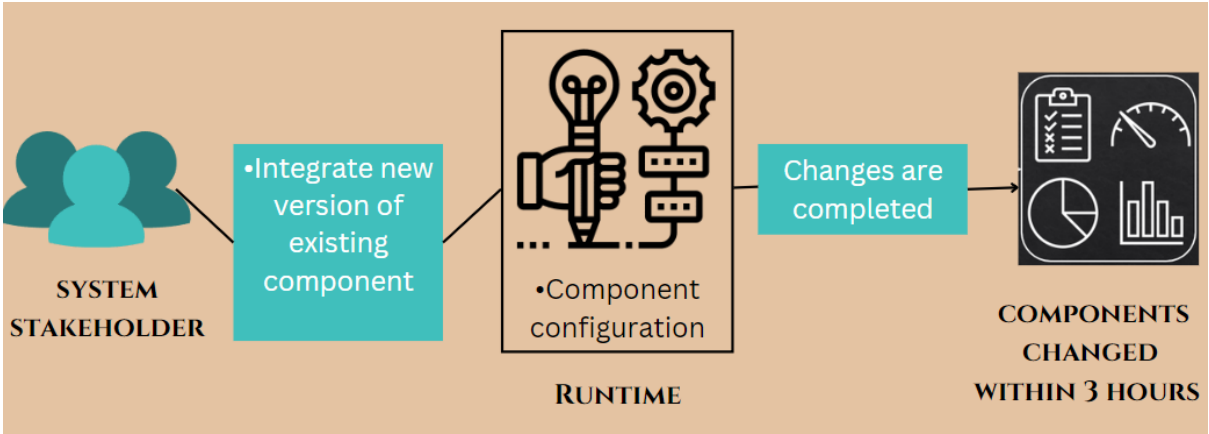


Quality Attribute Scenarios

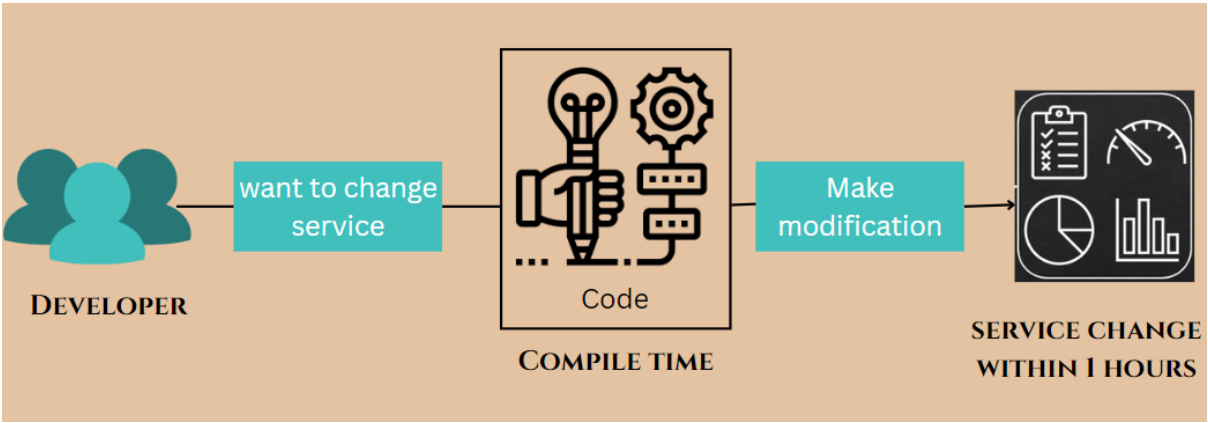
AVAILABILITY: Monitor



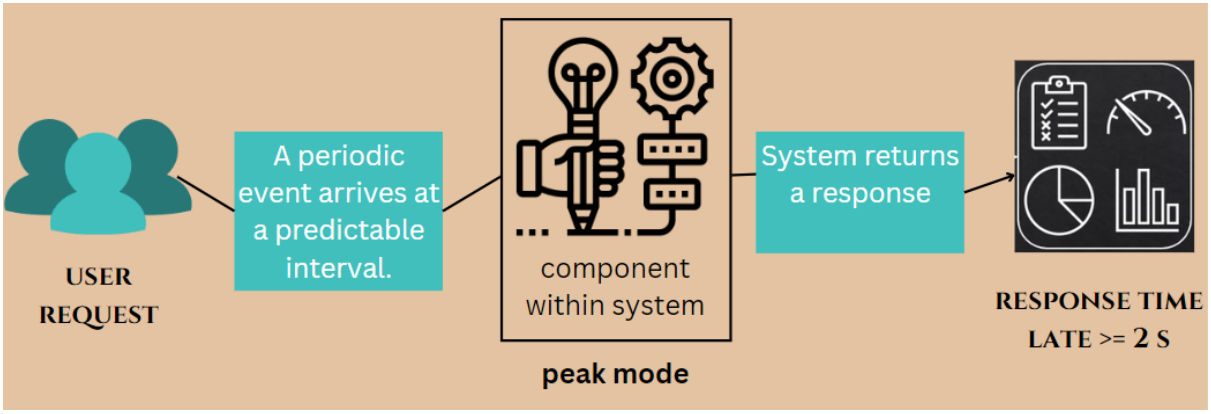
INTEGRABILITY : Configure Behavior



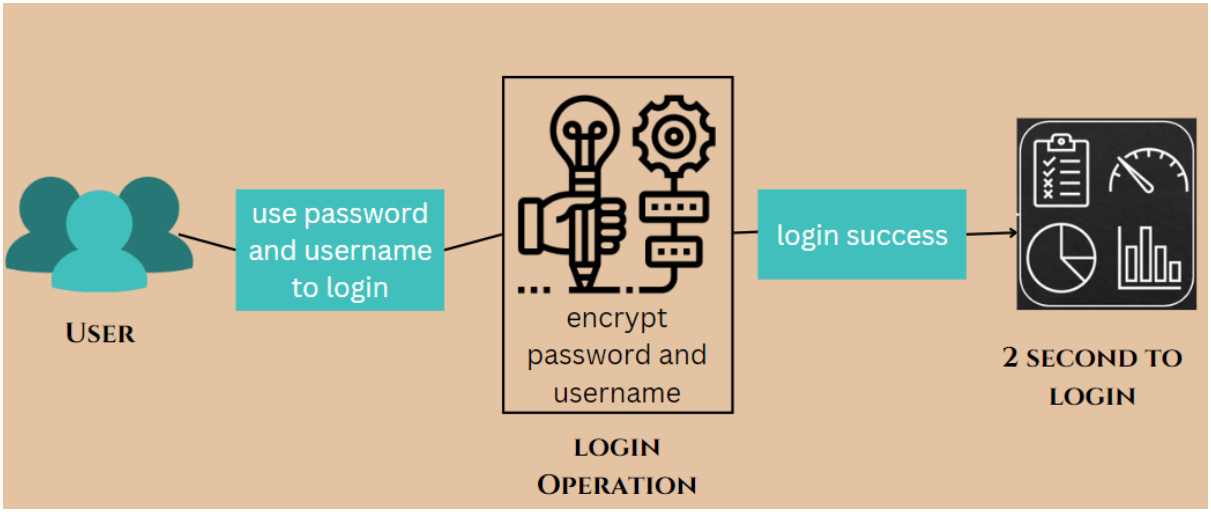
MODIFIABILITY :Split module



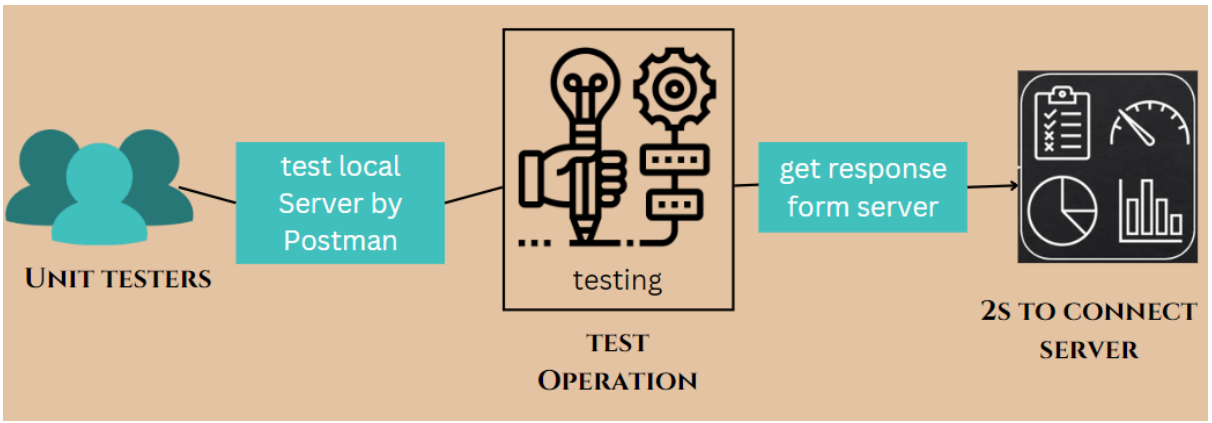
PERFORMANCE : Limit event response



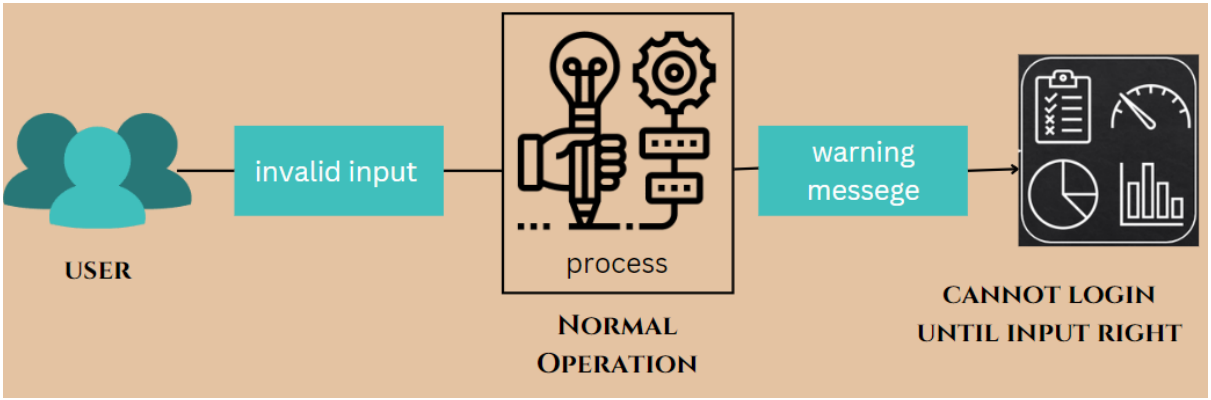
SECURITY : Encrypt data



TESTABILITY : Localize state storage



AVAILABILITY : Exception Detection



AVAILABILITY : Graceful Degradation