Producer Consumer Problem

Class A implements Runnable

{

Int capacity=100;

Int val=0;

@Override

Public void run()

{

While(true)

{

While(bucket.size()!=capacity)

{

Bucket.add(val++);

}

}

}

Class B implements Runnable

{

@OVERRIDE

Public void run()

{

If(list.size()==0)

{

}

Int val =list.remove();

Sopln(“element removed”+val);

}

}

class mAIN

{

List<Integer> bucket=new ArrayList<>();

Public static void main(String[] args)

{

A a =new A();

Thread t=new Thread(a);

t.start();

B b =new B();

Thread t1=new Thread(b);

t1.start();

Close

**33:20**

**Meeting chat**

**Subham Das (Guest)** has temporarily joined the chat.

Profile picture of Mallik, Atanu.

**Mallik, Atanu**

4:02 pm

Top of Form

Type a new message

Bottom of Form

Text

Description automatically generated

Controller layer

Admin Controller

Consumer Controller

Service Layer

Admin Service

Consumer Service

Repo Layer

Ticket Repository

Consumer Repository

AdminController.java

Class AdminController

{

@Autowired

Private AdminService adminService;

@RestController(“/tickets/adminService”)

Class AdminController

{

@Admin

@RequestMapping(“/addTicket”,method=RequestMethod.GET)

Public String addTicket()

{

Return adminService.addTicket();

}

}

@Service

Class AdminService

{

@Autowired

TicketRepository ticketRepo;

Public String addTicket()

{

Int countOfTickets= ticketRepo.getCurrentCountOfTickets();

Int newTicketId= countOfTickets+1;

Ticket ticket =new Ticket();

Ticket.setTicketId(newTicketId);

ticketRepo.save(ticket);

return “New Ticket created with id “+ticket.getTicketId();

}

}

Repository layer

@Reposoitory

Class TicketRepository implements CrudRepository<Ticket,Integer>

{

@Query(value=“select ticketCount from Ticket”,native=true)

Public void getCurrentCountOfTickets();

}

@RestController

@RequestMapping(“consumerService/consumeTicket

Class ConsumerController

{

@Autowired

Public ConsumerService consumerService;

@RequestMapping(“/consumeTicket”,method=RequestMethod.GET)

Public String consumeTicket()

{

Return consumerService.consumeTicket();

}