03/06/23

SDLC

- Req. Analysis ⇒ Planning —— Use case, BPMN
- Design —— ERD, Class Diagram, Interaction Diagram (Seq + Collaboration), State
- Development }
- Testing } → Continuous Integration, Continuous Delivery
- Deployment — Dev Ops — CI/CD pipeline
- Implementation
- Maintenance

Dev server – Test server – UAT/Staging – Production

BPMN → Business Process Modelling Notation

Good software

Req → Functional, Performance

- Functional req – completeness
- Non-functional (technical) req
  - performance → design, development } efficient → memory, computation
  - security
  - cost effective way তে resource utilization → normalized database
- Maintainability / Sustainability → target করলে S/W এর আপডেট version বাড়বে
- robustness / reliability → Code smell
- portability → Business Rule configuration / parameterization → similar type of client এর অনেককেই একটু অল্প change একই serve করতে পারি
- Usability → অশিক্ষিত মানুষেরও ইউজেবল করে design করা

Mock UI

↳ user feedback নেয়ার জন্য করা হয়

↳ user কে visualize করানোর জন্য

04/06/23

Requirement Engineering

→ iterative process

feasibility — technically feasible কিনা
budget এবং schedule এর মধ্যে feasible কিনা

authentication
authorization → authenticated user system এ যেসব কাজ করতে পারবে

availability → system টি up and running কিনা

reliable → more probably সঠিক এবং reliability যেমন হওয়া উচিত

slide-19

→ tax rate needs to be user-defined, not predefined
→ user purchase এ কি same rate? product-to-product vary করতে পারে?

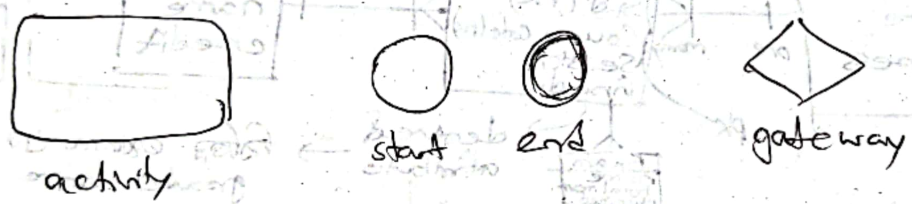
3) availability কত percent জানি জানতে হবে
কী কী security measure নেয়া উচিত, reliability কেমন হবে

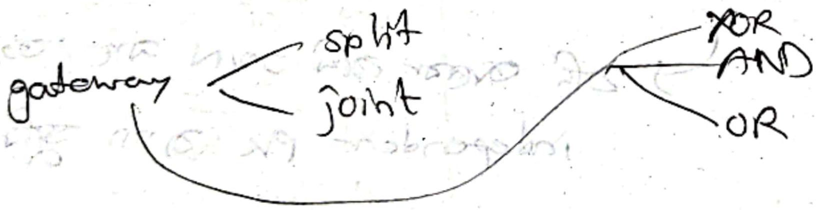
4) maintenance team → in-house
→ সেই অনুযায়ী criteria

6) requirement time এ তত detail এ যাওয়ার দরকার নেই

05/06/23

BPMN





activity → imperative, active

start event, end event → passive voice

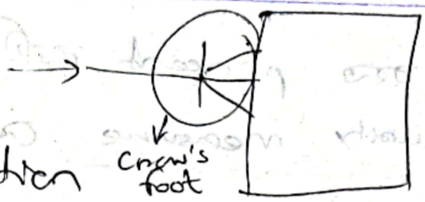
XOR joint → যেকোনো একটি true হলেই output triggered.

AND joint → সবগুলোই true হতে হবে

AND split → সব split activity-ই একসাথে হবে

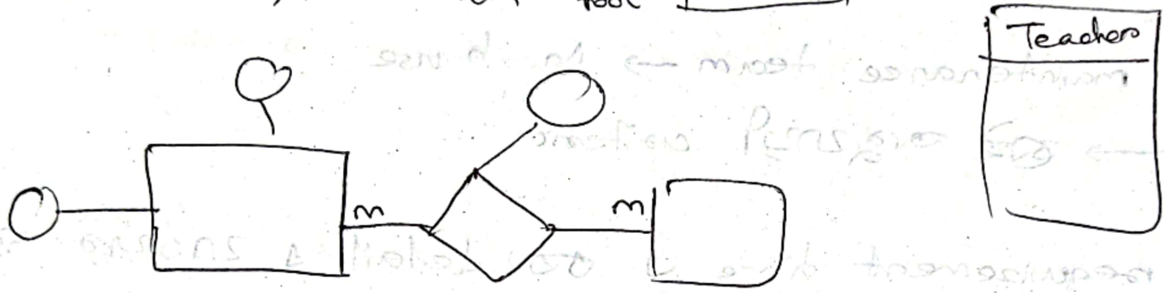
11/06/23

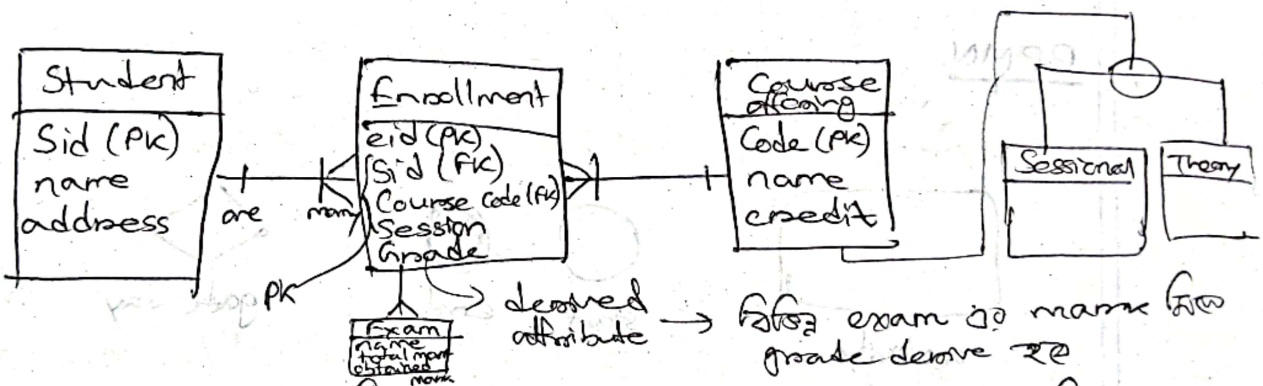
Crow's foot model →



Change model?

many to many relation





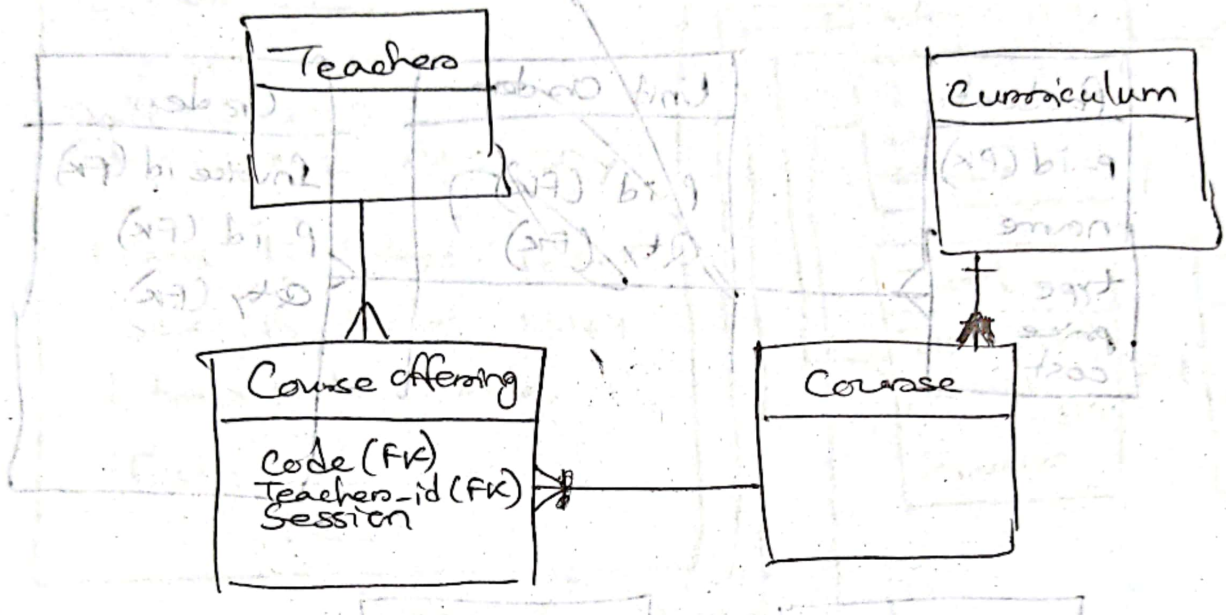
derived attribute → বিভিন্ন exam এর mark দিয়ে grade derive হবে

কারো independent PK দিবো আর কারো composite PK দিবো?

↳ যদি অনেক বেশি JOIN লাগে, তবে independent PK রাখা সুবিধা

eid → strong e

GradeRule	
from	
To	
from_mark	70
to_mark	75
grade	A



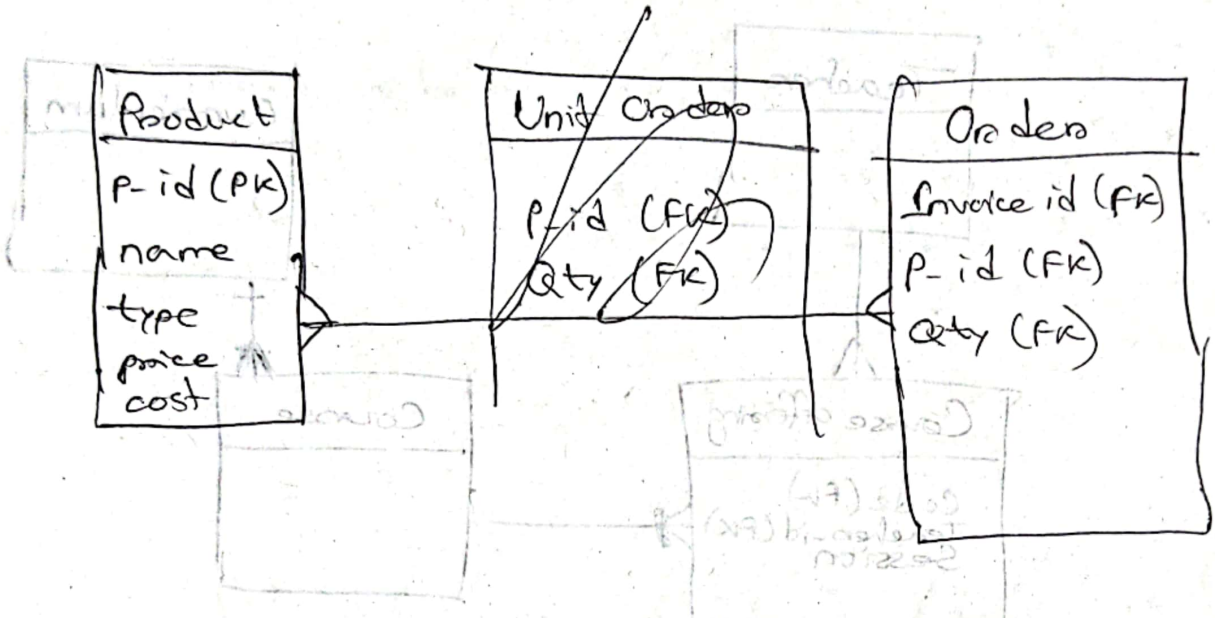
Invoice

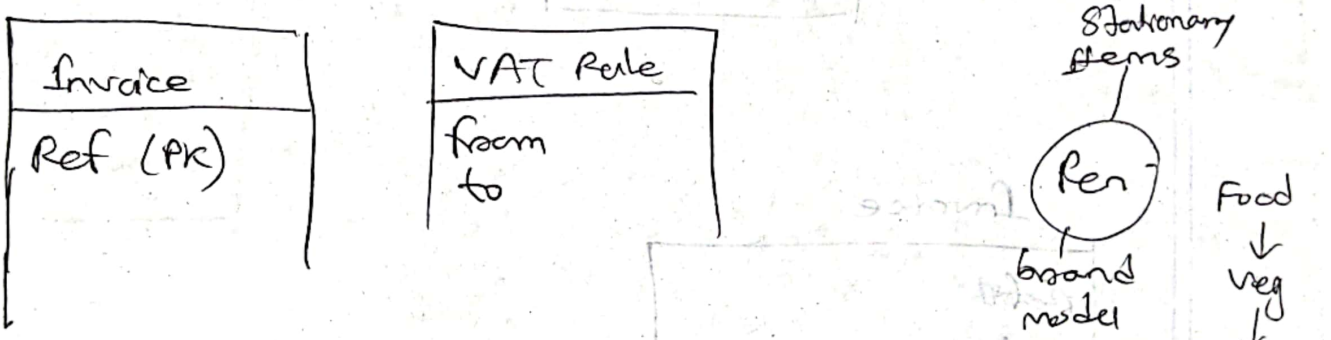
Ref #

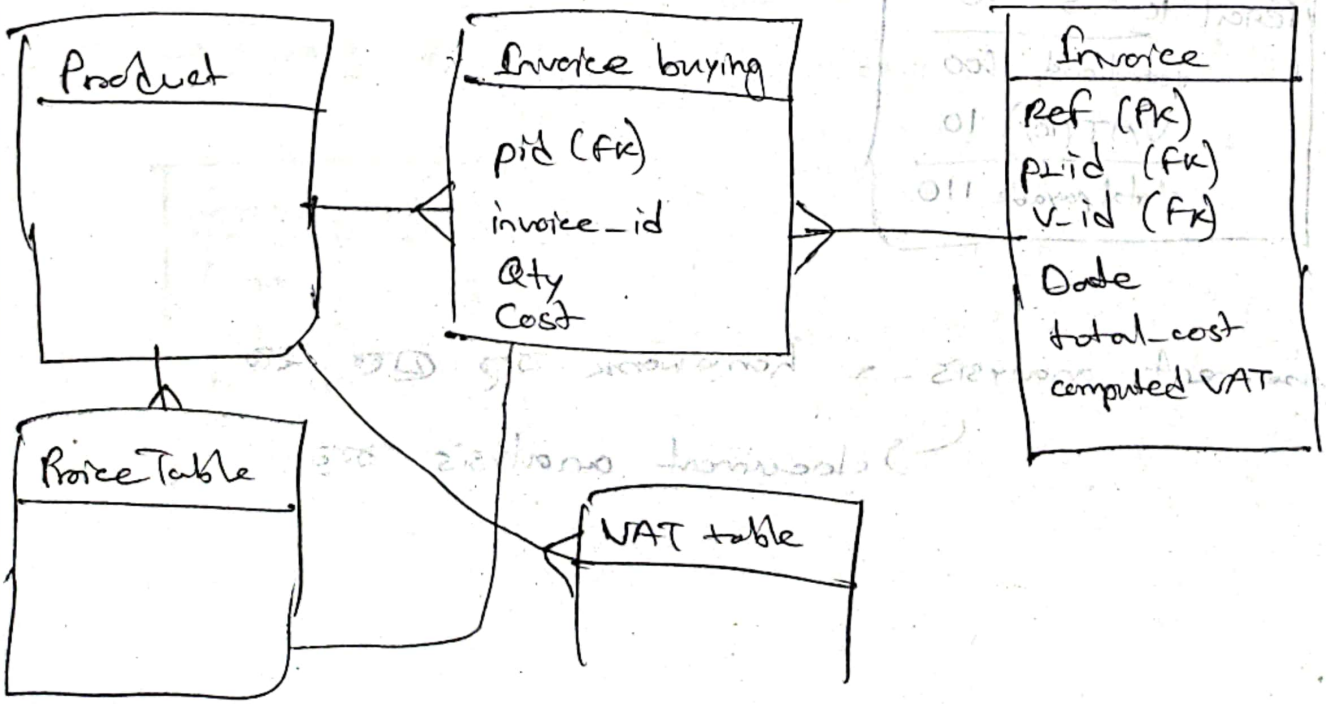
Date	Product	Qty	Unit Price	Cost
	Pen	5	10	50
	Pencil	10	5	50
			total cost	100
			VAT (10%)	10
			total payable	110

Requirement analysis → homework এর মত হবে

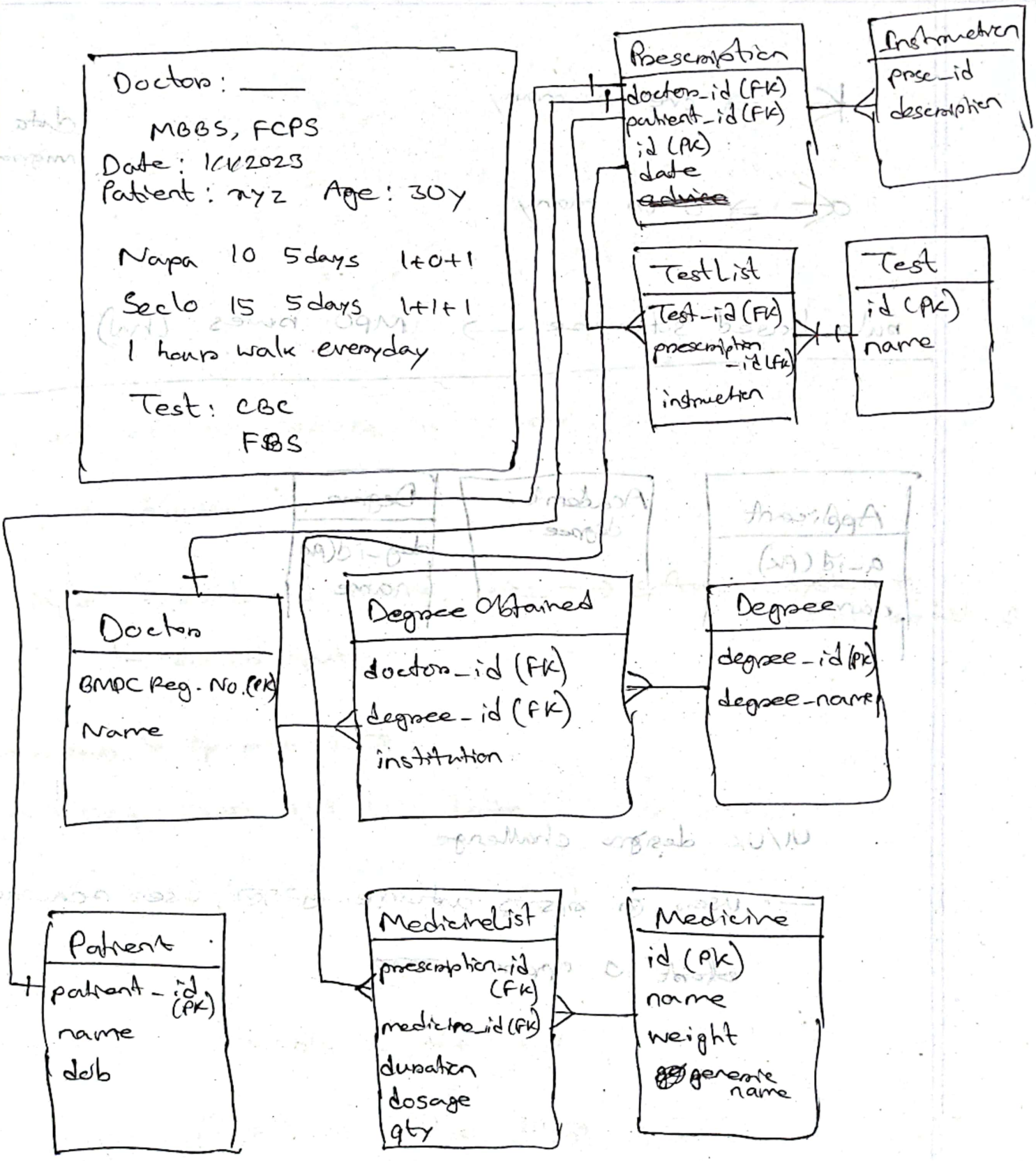
↳ document analysis হবে


Product
P-id (PK)
name
type
price
cost
Unit Orders
P-id (FK)
Qty (FK)
Orders
Invoice id (FK)
P-id (FK)
Qty (FK)


Invoice
Ref (PK)
VAT Rate
from
to
Stationary items
Pen
brand
model
Food
Veg
Tomato


Product
Invoice buying
pid (FK)
invoice_id
Qty
Cost
Invoice
Ref (PK)
P-id (FK)
V-id (FK)
Date
total_cost
computed VAT
PriceTable
VAT table

12/06/23

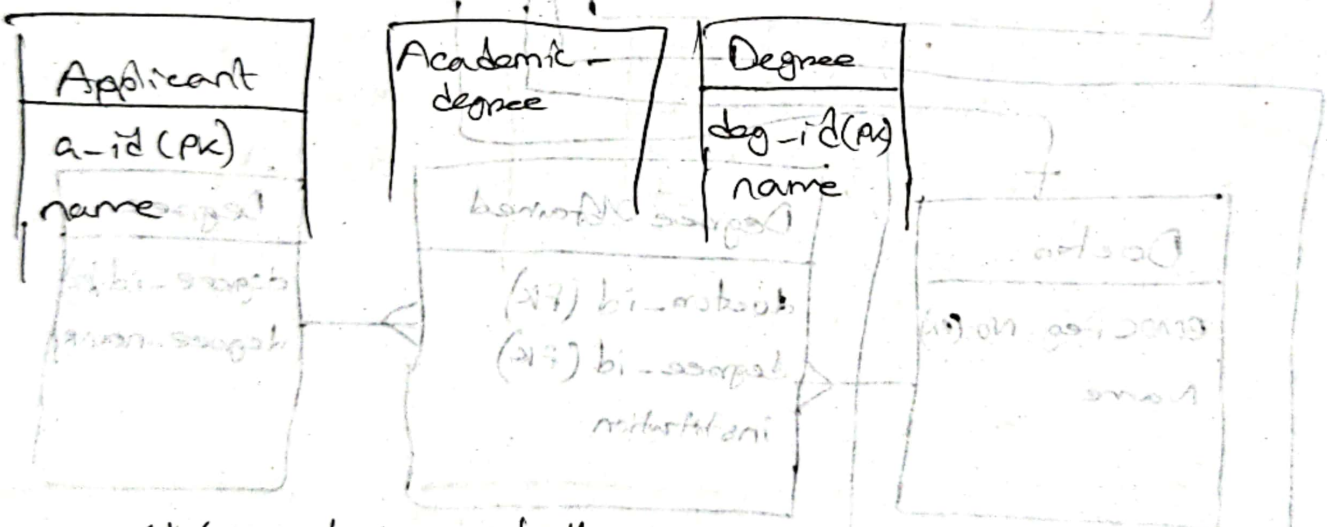

Doctor: ____
MBBS, FCPS
Date: 1/1/2023
Patient: xyz Age: 30y
Napa 10 5days 1+0+1
Seclo 15 5days 1+1+1
1 hours walk everyday
Test: CBC
FBS
Prescription
doctor_id (FK)
patient_id (FK)
id (PK)
date
advice
Instruction
prsc_id
description
TestList
Test_id (FK)
prescription_id (FK)
instruction
Test
id (PK)
name
Doctor
BMDC Reg. No. (PK)
Name
Degree Obtained
doctor_id (FK)
degree_id (FK)
institution
Degree
degree_id (PK)
degree_name
Patient
patient_id (PK)
name
dob
MedicineList
prescription_id (FK)
medicine_id (FK)
duration
dosage
qty
Medicine
id (PK)
name
weight
generic name

⟸ ⇒ one or many

⟽ ⇒ 0 or many

data migration

rule based software ⟶ MPU rules (hw)



UI/UX design challenge

⟶ user ⓐ কীভাবে extreme চেয়ে user reasonable

effort ও approve করে

State Diagram

Registration object

transition: a / b ← post-condition
↑
precondition

17/07/23

- Qualification requirement for applying for a job
  - academic, training
  - experience / age

22/07/23

MPO শ্রেণিবিভাগ

Entity

post_subject → উচ্চ বিদ্যালয়, মহাবিদ্যালয়, ...

academic_subject → পদার্থবিজ্ঞান, রসায়ন, ...

DevOps

Waterfall Model

ধ্রুবক pipeline ও server

→ dev server

→ developer-এর থেকে commit করে

→ github, ...

Dev → Test → Staging → UAT → Production

↓

Smoke testing → ডিপ্লয়মেন্ট run করে কিনা

→ test server

→ ইচ্ছাকৃত input ও fail করতে পারে

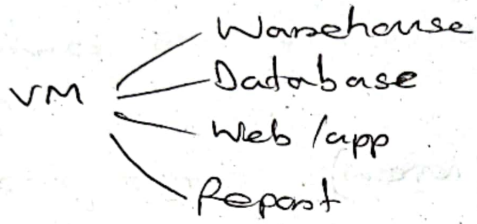
Staging → Production server এর replica

→ real-world এর ~~সঙ্গে~~ ডিপ্লয় করা

→ stage rehearsal type

UAT → users কে test করতে দেয়া

Production server



→ client ও requirement change হয়

→ environment change হয়

Agile Method

~~Stay~~

CI/CD → Continuous Integration
Continuous Delivery

containerization / orchestration → resource নিয়ন্ত্রণ / manage

Selenium → UI - based testing

23/07/23

Non-functional requirement

MVC architecture

Boundary class → V (View)
Controller class → C
Model class → M

constraint
- → performance constraint } can be conflicting
- → security constraint

↓ security measure বাড়ালে perfomance sacrifice করতে হতে পারে

Data-centered architecture

e.g. NID database, birth reg db, educational db

component → independently কাজ করতে পারে
→ অনেক req efficiently handle করতে পারতে হবে
data এক জায়গায় → manage করা easy

→ but ওই data fail করলে?

→ DRS রাখতে হবে

→ performance খারাপ হতে পারে
→ অনেক req এক জায়গায়

→ hardware resource enhance করতে হবে

→ cost বাড়বে

~~ore~~ warm backup
→ hot আর cold এর মাঝামাঝি

DRS
→ Disaster Recovery System

hot swapping / hot backup
→ প্রতি মুহূর্তে backup রাখা

cold backup
→ particular period পরপর backup

client - server

→ বিভিন্ন service কে আলাদা আলাদা server দিয়ে serve করতে পারি

→ server গুলো কাছাকাছি থাকতে not necessarily

→ distributed

→ geographically distributed

↓

shared db ?

CDN

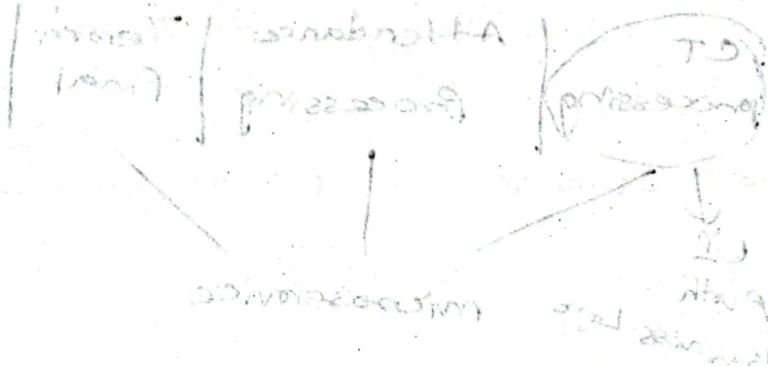
↓

Content Delivery Network

→ একটা service এ শুধু single point of failure

→ e.g. payment server down ⇒ সব service payment service বন্ধ

→ dependency on ই network



~~24/07/~~
24/07/23

MVC Architecture

→ client server থেকে ~~কোন~~ কোন service ও চাইলে MVC arch. follow করা যাবে

routing class

multiple ways to view — mobile app
web app

application specific logic → email validation etc.

→ controller layer থেকে তথ্য নিয়ে service layer ও দিয়ে যাওয়া

↓

যে কোন ধরনের computation করা

business logic → tax calculation

Result Processing ← Service

CT processing | Attendance Processing | Term Final

↓

UI
Auth
Business Logic

microservice

- প্রত্যেক ~~mic~~ microservice এর db আলাদা আলাদা করা

→ μ-service এর through ই db interact করবে
→ অন্য কোন inter-service communication নাই
→ team গুলো independently কাজ করবে
→ focused way তে কাজ করা যাবে
→ quality ↑

19/08/23

Software Security

exploit → script / program যেটি use হয় কোন একটি system এর vulnerability কে exploit করার জন্য

confidentiality → sensitive data কে unauthorised access থেকে রক্ষা করা

unauthorized e.g. student marks in BUS
new access

integrity → purity of data

→ তথ্যকে কেউ alter করতে না

unauthorized editable access

availability → system যেন down না হয়ে যায়

requirement যেন ভাল h/w support না দিতে পারে

e.g. ddos attack / Dos attack

(distributed DoS)

static analysis → compiler এর মত code scanning

dynamic analysis → input এর basis এ scanning

Firewall
- layer 3/4 পর্যন্ত কাজ করে
- Application level firewall & e.g. WAF → web application firewall

Non-repudiation → accountability → একজন কিছু করেছে ensure করবে যদি অস্বীকার

- ↳ transaction এর জন্য important
- ↳ strong authentication দরকার e.g. OTP

→ what you know → e.g. password } দুটি দিয়ে 2FA
→ what you have → e.g. phone

→ আরো security লাগলে → biometric authentication / hardware authentication

what you have ↗ H/W Dongol

আবার অন্য)
exclusive access use করে কিছু করেছি → যদি আমি অস্বীকার করতে পারবো না
→ responsibility আমার

privacy vs non-Repudiation

floating receipt

→ voting : privacy is more important

→ voter - ~~candidate~~ vote এর কোন mapping থাকা যাবে না

20/08/23

Identification
→ user ক identify করতে হবে

{ people
technology
process

authentication
→ user ক কী role দিবে

→ বন্ধুর bus কিনা, (bus partner different কিনা) এটির ওপর base করে secondary verification

→ requirement গুলোর risk analysis করতে হবে

anomaly detection by data ~~anal~~ science team

→ regulars pattern এর বাইরে abnormal behaviors

authorization
→ প্রতি role এর কতটুকু privilege থাকে

BYOD
↓
Bring Your
Own Device

→ user authentic users কিনা
→ এই particulars কাজের সবাই authorized কিনা

IT audit → IT এর recommended processing practice
track করে

violation of authorized activity

security bug থাকতেই পারে

→ ensure করতে হবে standard procedure follow হচ্ছে কি না

priviledge escalation attack

→ priviledge বাড়িয়ে নেওয়ার attempt

→ user এর যতটুকু দরকার, তার চেয়ে বেশি priviledge দেওয়া যাবে না

→ continuous testing এর মাধ্যমে ব্যবহার হবে

→ আগেরটা নাই দেখে পরেরটা ও হবে না তার guarantee নাই

21/08/23

client, server দুই side এই validate

→ client side এ check

→ ~~er~~ detect করলে ব্রাউজার থেকেই ফেরত

→ bandwidth consumption কমে যায়

~~OWASP?~~ ~~OWASP?~~ OWASP

3rd party থেকে ~~a~~ আসা data কে sanitize

→ attack হয় inside থেকে

→ zombie way

→ insider থাকবেই না ও

e.g. trojan

Logical Incorrect Query

→ ইচ্ছা করে syntax error তৈরি করে response দেখা

→ কী db use হচ্ছে, তার version কত তাও জানা

Timing attack

→ অনেক db অনেকক্ষণ delay handle করে

→ response ~~আসা~~ দেরি হলে বোঝা যে কোন db use হচ্ছে

@pple.com

↓

মানুষ যা ভাবছে

অন্য সাইট similar char use → যেমন 'a' তা মতো

→ user ভাবছে apple.com এ গেছে

profanity bypass → usually similar char দিয়ে replace

02/09/23

→ 1st step → data insert / update

2nd step → attack

ORM?
ORMN?
WARMN?

Integer overflow → memory ত garbage value ঢুকানো

→ system crash করে দিতে পারে

parameterized query

→ query - ? structure করে দেই

```
SELECT ...
FROM ...
WHERE user: ... AND pass: ...
```

→ এটি যে যেকোনো কোড আগেই structure টা define করে দিয়েছি

→ bypass করার আর সুযোগ নেই

static analysis → code analysis করে যেসব vulnerability আছে তিনি

→ e.g. query structure define না করেই query execute হচ্ছে

dynamic analysis → injection এর জন্য tiny tiny content generate করে

→ কোন যেসব আক্রমণ হচ্ছে কিনা

seq diagram

→ প্রতি use case এর জন্য আলাদা করে দেখি ছবি বানাতে হবে

[see Benett's book]

collab diagram → loop এর concept নেই

↳ arch এর রূপের কাছে

(layered arch)

→ এই এই actor /model আসা লাগবে সবই দেখাতে লাগবে

same diagram এ

03/09/23

Cross-site scripting → browser থেকে info collect

SQL injection → db থেকে info collect

cookie → server এ একটা user identity create করে রাখে

→ user log in না করলেও user কে চিনতে পারে

cookie থেকে info collect করে user হিসেবে pretend করতে পারে

→ say, prothom alo site এ আমার cookie saved

→ এই cookie info দিয়ে অন্যজন prothom alo এ + দুনিয়া ঘুরতে পারে আমাকে কী কী ad দেখানো হয় ⇒ আমার পছন্দ কী দিচ্ছে → breach of privacy

XSS (3 types)

→ stored / persistent
→ reflected
→ ~~[next class]~~ DOM-based

→ stored
  → যেই site এ ঢুকছি, ঐ site এর মধ্যেই attacker script লেখা দিয়েছে
  - e.g. comment in a blog → যে কেউ comment করতে পারে → site এর মধ্যেই exist করছে
    → attacker comment এ link save করে রেখেছে
    → আমি ঐ comment এ click করলে ঐ link এ নিয়ে যাবে
      → link টি basically একটি js script → যেটি run হয়ে যাবে

→ session id → বারবার session create করা → server এর জন্যও expensive, client এর জন্যও বিরক্তিকর
  → session id নিয়ে গেলে আমার হয়ে access করে ফেলতে পারবে

→ man in the middle attack
  → non-https site → pckt capture করেই নেওয়া যায়

→ comment sanitize করা
  → comment কে db তে direct store না করে malicious content আছে কিনা check

reflected

→ victim নিজেই initiate করে

e.g. phishing link

→ http req. এ সংযুক্ত script browser এ run করে

~~→ web~~

04/09/23

→ website এ কিছু করার নাই

→ separate link এ করা হচ্ছে

→ website sanitize করতে পারছে না, করলেও তা ভালো না

→ attackers social engineering করে প্রথমে user কে target করে

reflected xss → CSRF Cross Site Request Forgery

→ server side এ হয়

→ server side log এ trace পাওয়া যায় investigation time এ

DOM-based xss

→ js ভিত্তিক based execution এ সংযুক্ত attack হয়

→ server side এ কোন log trace হয় না

→ why?

~~pre~~ modify presentation of content → can lead to defamation, create chaos, spread rumours/false news

09/09/23

Microservice Arch

Say, inventory is a service

new approval
requisition } μ-service

→ common use এর db এর উপর base করে μ-service design

→ প্রত্যেক μ-service এর db আলাদা

operational capability → ~~service management~~ প্রত্যেক service কে নিয়ে

management .. → কোন service কে scale করবো

CDN → Content Delivery Network

→ distributed system of proxy servers

↳ network এ bandwidth consumption কমানো

contextualized recommendation

→ ~~কি~~ আমরা কিছু recommend করার আগে, how to shortlist

→ user যা চেয়েছে, তাই যদি দেয়, তার প্রতিক্রিয়া কী হবে

→ μservice design এর প্রথম কাজ use case নিয়ে explain

Message Queue

Rapid MQ

Kafka

inter-service communication এ ব্যবহার করা queue maintain

Caching

Redis

read-heavy operation

write-heavy operation

db-indexing

→ যেই table এ write বেশি, সেখানে index কম থাকা ভালো

→ শুধু id এর উপর indexing

→ but, say নাম এর উপর indexing করা

→ write operation অনেক slow হয়ে যাবে

→ প্রতি write এ re-index করা লাগবে

→ যেখানে search বেশি

→ যেই param এর উপর search বেশি → indexing করলে performance ভালো হয়

10/09/23

Performance Optimization Technique

performance → load endure করার capacity

engineering optimization → for a given resource, performance সর্বোচ্চ improve করা হয়

Message Queue

→ দুটো sw component এর মধ্যে communication → মাঝে queue থাকবে
→ একটা sender
→ এক/একাধিক receivers

sms, pdf to doc converters website

→ async communication
→ persistent queue maintained by web servers
→ users এ ফাইলে এ file receive হচ্ছে (status)
→ যখন backend servers free হবে, তখন web servers queue থেকে নিয়ে backend এ পাঠাবে
→ ~~pe web~~ web servers এ req অনেক হলো, কিন্তু backend slavers কম, message queue দিয়ে এই load maintain করা যাবে
→ multiple component/application কে decouple করতে help করে

→ allows traffic spike
  → additional traffic কে queue তে রেখে দিল → response slow হবে, but পাওয়া যাবে
→ independently scale করা যাবে
  → user কে ও জানানো হবে যে req received, তবে কিছু সময় লাগবে
  → say queue তে 100 req
    server আছে 5টি
    → ~~ser~~ backend scale করে server বাড়িয়ে দিলাম
      → user খুশি
      → at least status তো জানতে পারছে
→ bash processing

Distributed Caching

newspaper website → read-heavy

database log → write-heavy

instant messaging → read + write

application এর property vary করে in terms of read and write frequency

→ read, write এর frequency -র উপর base করে আলাদা আলাদা technology

cache কতদিন রাখবো

→ say, a news
  → একদিন cache রাখলেই enough

→ movie
  → release থেকে 1 month পর্যন্ত cache রাখলাম
  → content change হয় না after ~~years~~ release

LRU → not always good

cache এ performance কেমন
→ regularly monitor করো
→ হতে পারে unnecessarily cache overhead হয়ে যাচ্ছে

<u>cache aside</u>

db থেকে read করে app থেকে cache এ write এর মধ্যবর্তী time

যদি লাগে → application এ cache miss হয়ে যায়

→ সরাসরি db থেকেই cache update
  → read-through
  → এখন এ application এ req করছে কি additional benefit পাবে না
  → এভাবেই cached data পাবে

lazy

→ req না আসলে মূল মূল তথ্য থেকেই cache update হচ্ছে না

11/09/23

write-around

→ write এ performance improve হয়

→ cache এ দুর্বল হয়

→ in combination with cache-aside

→ log file বা এজাতীয় কাজে

write-back

→ write-heavy

→ db তে দিতে আসলে → latency বাড়ে

→ cache → sufficiently large হতে হবে

load balancing

→ response time এর উপর base করে

→ যেই server বেশি fast respond করছে তাকে assign

→ bandwidth balance

→ একজনকে 1000 unit, আরেকজনকে 500 unit

hardware sizing

vendor - neutral design

F5
↓
load balancer

Layer 4

Layer 7

high availability → একটি server down হলেও এর handle করা যায়

↳ নিজস্ব cache → ওই server কে যা পাঠানো হয়েছিল সেটি আরেকজনকে পাঠানো

multiple load balancer রাখা