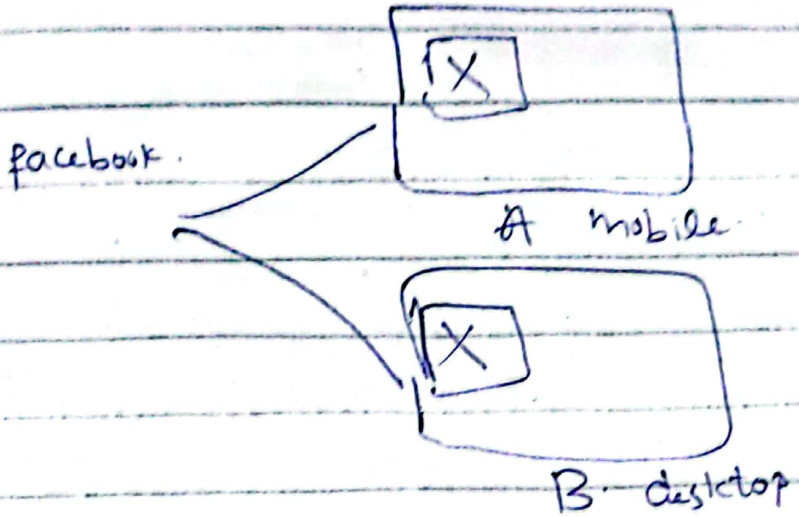
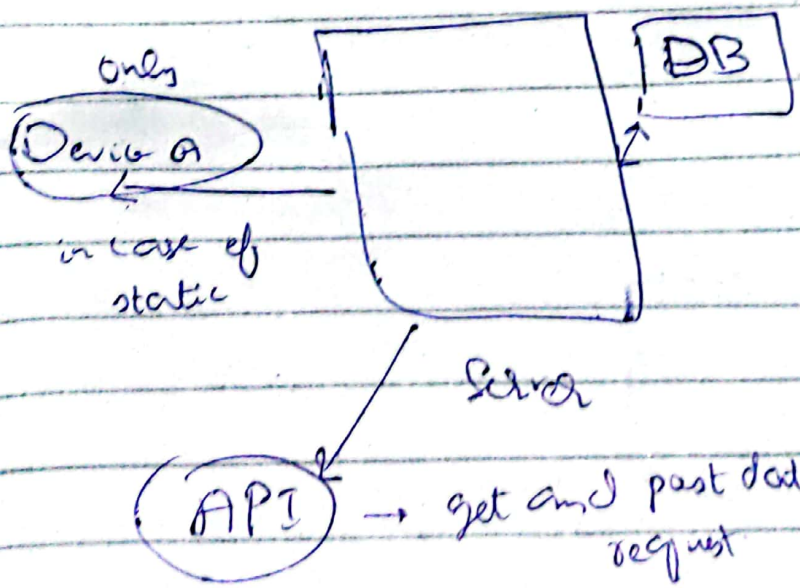
REST API

website → static design

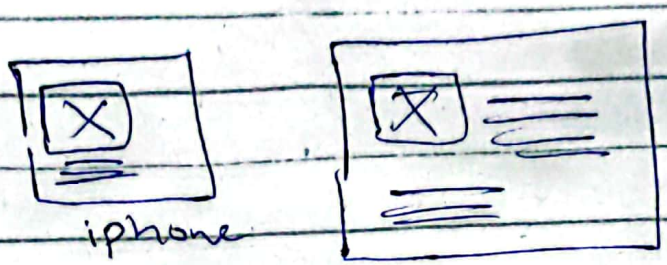
Now, website → dynamic design





Three types of API

(i) Soap api (xml)
(ii) Rest api (json)
(iii) .NET API (asp).



all changes are reflected:
API support all the devices
to manage data dynamically

API: Post, PUT, GET,
~~READ~~, DELETE

- JSON (all languages can understand).

- GSON.

ML.

↓

pickel. (file).

↳ deploy → free service.

↕

API

flask.

Todays:

(i) Install postman.

(ii) Android Manifest.xml.

↳ add internet permission.

(iii) Google properties (module).

↓

add implementation (resto).

Lecture 9

Week 10

Mobile Application Dev

(MAD)

1. Add permission in manifest xml file.
   internet permission
2. Use REST API.
   (retrofit android dependency)
   add dependency lines in gradle file (app module).
   - Retrofit
   - GSON
3. GSON
   - convert Java object into JSON object and soon.
4. Add java class (RetrofitInstance).

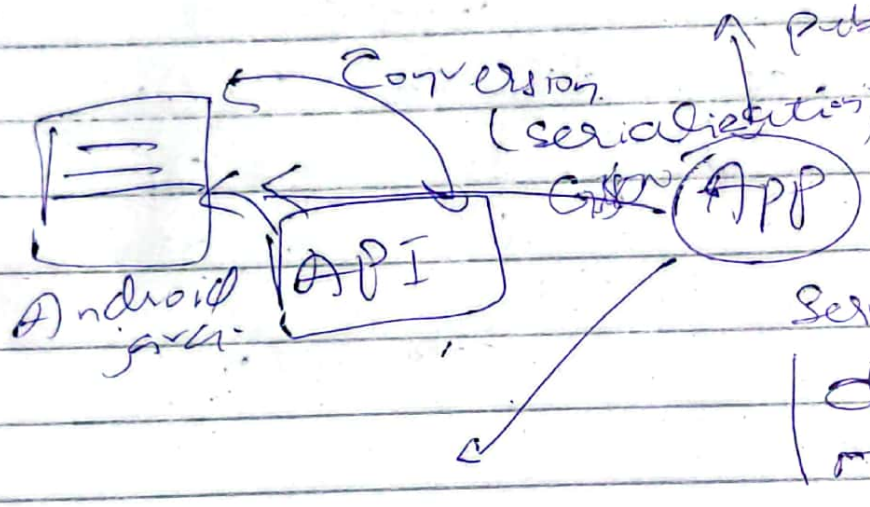
(3) · Create Java class.

↳ Retrofit Instance.

↳ static object of Retrofit (Imported from)

↳ base url (used for annotations).

(Common URL)



(4) · Create Model Post Class (model) with all getter and setter function.

5. Create Interface class in project (interface for endpoints)

@GET("posts")

call<List<ModelPosts>> getPosts();

6.

Date:

24-Oct-23

Tools → Firebase

Firebase → Login → Add Project → Automatically links to your project

Configure google analytics

Realtime database
→ Implementation mein version hata dena hai.

6- Nov-2023

(REST API) → dynamic
→ 3 types of API
→ more flexibility
→ faster than soap API
→ less security than "
→ multiple formats

xml

Multiple formats mein aate hi cheez.
API is a helper for all (desktop, android, IOS)
→ Middleware (developer uses this API)
→ using sqlite, changes are only made to your device as shown of yours only.

text → obj deserialization
obj → text serialization

ML → Pickil → file → deploy
↑ API flask app

Install POSTMAN

→ create project
→ add permissions (internet, retrofit)

Date:

17-Nov-23

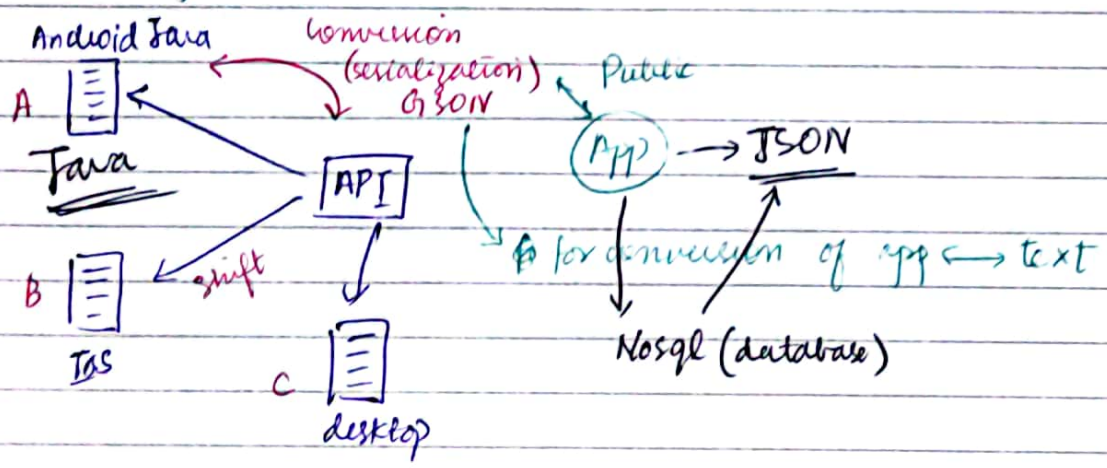
difference in language → bridge, communication is done by API

Java make class → Retrofit instance
↳ create object

website: JSON2CSharp

1- Internet permission in Manifest file
2. Add dependencies in gradle file (app.module) → Retrofit, → GSON
↳ SYNC GRADLE
GSON: serialization, deserialization

3. Create java class for retrofit instance (retrofit class)
↳ static object of retrofit (Implicit dependency)
→ Base URL (used for annotations) (common url)
use a facility



4. Create Modal class (Use Json to Java Class(object) Converter) → use same name
↳ Generate getter setter of Java class (automatic)
→ relating java obj. to JSON
New → Java class → interface
write call<List<ModelPost>> getPost();

copy from model paste to AS. (create getter setter)

Date:

5. Create Interface for (endpoints) API
→ GET Method (to get the list of posts from Base URL acc to the defined Model class)
* use list method

```
apiInterface = RetrofitInstance.getRetrofit()
                .create(ApiInterface class),
```

* API → object (2 overridden methods.)
failure successful.

[For ML Model]
import pickle
↳ download the pickle file.