**What is an API?**

* API Sstands for Application Programming Interface
* It acts as a two-way communication bridhge between frontend and backend.

# Client → HTTP Request (REST API) → Server → HTTP Response (REST API) → Client

**What is REST API?**

* REST stands for Representational State Transfer.
* It organizes how web applications talk to each other, separating what the user sees (frontend) from what runs behind the scenes (backend).

**Core Principles of REST:**

* Stateless: The server doesn’t store any information about the client between requests.
* Client-Server Architecture: The app (client) asks for things (data) and the server does what’s requested (sends data or makes changes)
* Standardized Interface: REST APIs rely on a set of standard methods (POST, GET, PUT, PATCH, DELETE) for interacting with resources.
  + POST: Creating the resource
  + GET: Retrieving the resource
  + PUT: Update the entire resource
  + PATCH: Update the partial resource
  + DELETE: Deleting the resource
* Easy-to-Read Data: REST APIs returns the response in a standardized easy to read formats, typically JSON or XML formats.

**Web Application Endpoints:**

* Web Application Endpoints: Users can directly access it from the web browsers (**Returns HTTPResponse**)
  + <http://127.0.0.1:8000/students>
* API Endpoints: Returns the data to integrate into the frontend (**Returns JSONResponse**)
  + <http://127.0.0.1:8000/api/v1/students/>

**How to access APIs hosted in same wifi?**

* ip addr show | grep inet : Something will be written like this → inet 192.168.1.3/24 brd 192.168.1.255 scope global dynamic noprefixroute **wlan0 →** It is the **Local Wi-Fi IP**
* Add this IP to django\_rest\_main → settings.py → ALLOWED\_HOSTS = [‘192.168.1.3’]
* Then run the server with this: python manage.py runserver 0.0.0.0:8000 → This tells Django to listen to all Ips (not just 127.0.0.1)

**Serializers: (Translator between Django Objects and JSON Data)**

* It converts complex data types (like Django models or querysets) into JSON (or XML) that can be sent to the frontend or API clients.
* It can also desrialize JSON data back into Django Objects.

Why we need Serializers?

* Django models or querysets are not JSON serializable by default.
* Frontend or API clients expect JSON Data.
* Thats where serializers come in → they prepare data for transmission and validate incoming data.

**Class Based Views:**

* It provides more structured and organized way to handle requests using Object-Oriented Principles.

**Mixins: class Employees(mixins, generics.GenericAPIView)**

* ListModelMixin → list()
* CreateModelMixin → create()
* RetrieveModelMixin → retrieve()
* UpdateModelMixin → update()
* DestroyModelMixin → destroy()

**Frontend Installation:**

* npm create [vite@latest](mailto:vite@latest)
* Project Name: stock-prediction-portal
* Select a framework: React
* Select a variant: JavaScript
* Use rolldown-vite (experimental) : No
* Install with npm and start now?: No
* Move all files into frontend folder and delete the Project Folder (stock-prediction-portal)
* And run npm install
* npm run dev → To run the development server
  + To add Tailwind CSS:
    - npm install -D tailwindcss postcss autoprefixer
  + Search in google → **cdn bootstrap 5 (To add inbuilt CSS & JS Support)**
  + Go to → <https://getbootstrap.com/docs/5.0/getting-started/introduction/>
  + **To add inbuilt CSS: (Bootstrap)**
    - Copy CSS → <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet">
    - Paste inside index.html → inside <head></head>
  + **To add JS Support: (Removed integrity and crossorigin)**
    - Copy <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"></script>
    - Paste it index.html → inside <body></body>
  + **To add Custom CSS for our project:**
    - Create src/assets/css/style.css
    - In App.jsx → import ‘./assets/css/style.css’

### DJANGO Initialization:

* mkdir -p Stock-Prediction-Django-React-ML
* cd Stock-Prediction-Django-React-ML && python -m venv venv
* source venv/bin/activate
* mkdir -p backend
* mkdir -p frontend
  + cd backend
  + pip install django

### DJANGO Secret\_Key generate:

* Go to <https://djecrety.ir/>
* Generate and Move it to .env: SECRET\_KEY=d^-yvq1hom+zue^\_glf(nhcs(r$h$r10eyc21ix$lr@)l(k\*+l
* pip install python-decouple
* In stock-prediction-main → settings.py → Remove existing SECRET\_KEY
* Add this line: config(“SECRET\_KEY”)
* Import library: from decouple import config

**DJANGO Debug in .env:**

* In settings.py → DEBUG=config(“DEBUG”, default=False, cast=bool)
* In .env: DEBUG=True

**Setup Github Repo:**

* To generate .gitignore file:
  + Go to Google → Search for generate gitignore file for project
  + Click <https://www.toptal.com/developers/gitignore> → Select Language (Django, React)
  + Copy all lines to root/.gitignore
* Now go to Github.com
  + Remote
    - Create repo → Stock Prediction Portal → Create repo
  + Local
    - git init
    - git add .
    - Git commit -m “first commit”
    - git branch -M main
    - git remote add origin [git@](mailto:git@github.com)github-personal:sm60786/Stock-Prediction-Portal.git
    - git remote -v
    - git pull origin main –rebase → As I created README.md file in cloud
    - git push -u origin main (First Commit/Push Done)