

# AI-Based Math Tutor - Complete Project Plan

## 1. Project Idea

- Build an AI-based Math Tutor as a web app.
- Focus on step-by-step problem solving, visualization, and user progress tracking.
- Main goal: modern, interactive web app without deep diving into frontend, keeping AI/ML/Data Science as the core.

## 2. Tech Stack

- Backend: Python + Flask (or FastAPI later)
- Math Engine: Sympy, NumPy, Pandas
- Frontend: HTML, CSS, Bootstrap, MathJax, minimal JS (fetch API)
- Database: SQLite
- Visualization: Matplotlib (static) -> Plotly (interactive graphs)

## 3. Learning Roadmap (Phase-wise)

### Phase 0 – Python Refresh (3-4 days)

- Revise functions, OOP basics, error handling
- Mini-Project: `EquationSolver` class for linear equations

### Phase 1 – Backend Basics (Flask) (1 week)

- Install Flask, create routes (`/`, `/solve`)
- Handle JSON input/output
- Mini-Project: "Hello Math Tutor" API

### Phase 2 – Math Engine (Sympy) (1-2 weeks)

- Solve linear and quadratic equations
- Differentiation & Integration
- Step-by-step solution formatting
- Mini-Project: `solver.py` with functions `solve_linear`, `solve_quadratic`, `differentiate`, `integrate`

### Phase 3 – Frontend Basics (1 week)

- HTML form input + Bootstrap styling
- MathJax for equation rendering
- JS fetch API to communicate with backend
- Mini-Project: `index.html` interactive page

## Phase 4 – Database (SQLite) (1 week)

- Setup DB with `Users`, `Problems`, `Scores`
- CRUD operations with Flask
- Mini-Project: `db_setup.py` for tracking user progress

## Phase 5 – Visualization (1-2 weeks)

- Plot equations with Matplotlib -> display in web app
- Upgrade to interactive Plotly graphs
- Mini-Project: Plot  $y=x^2+2x+1$  and show in browser

## Phase 6 – Extras / Advanced (Optional)

- Quiz mode & hints
- Login/Signup system (Flask-Login)
- Speech-to-text & Handwriting recognition
- Modern React.js frontend
- Upgrade backend to FastAPI
- Mini-Projects: implement each optional feature individually

## 4. Mini-Project Plan (1-2 days each)

| Phase | Mini-Project                  | Goal                           |
|-------|-------------------------------|--------------------------------|
| 0     | Linear Equation Solver        | Practice modular Python & OOP  |
| 0     | Simple Utility Script         | Input/output & error handling  |
| 1     | Hello Flask App               | Understand basic Flask routing |
| 1     | JSON Input API                | Test API request/response      |
| 2     | Linear & Quadratic Solver     | Sympy problem solving          |
| 2     | Differentiation & Integration | Build AI math brain            |
| 3     | Input Form                    | User interface basics          |
| 3     | Display Solution              | Connect frontend ↔ backend     |
| 3     | Bootstrap Styling             | Modern look                    |
| 4     | Database Setup                | Store users & problems         |
| 4     | CRUD Operations               | Track user progress            |
| 5     | Matplotlib Graph              | Visualize equations            |
| 5     | Display Graph in Web          | Integrate graph with UI        |
| 6     | Quiz Mode                     | Interactive problem-solving    |

| Phase | Mini-Project             | Goal                     |
|-------|--------------------------|--------------------------|
| 6     | Login/Signup             | User authentication      |
| 6     | Speech/Handwriting Input | Advanced input methods   |
| 6     | React Frontend           | Polished UI              |
| 6     | FastAPI Upgrade          | Async & scalable backend |

## 5. How to Use This Plan

1. Follow phases sequentially -> build confidence while learning.
2. Complete mini-projects per phase -> small wins every 1-2 days.
3. Integrate all mini-projects into full AI Math Tutor by Phase 5.
4. Optional Phase 6 -> add advanced features once core app works.
5. Keep checking off milestones -> track progress visually.