Clinical Lab Companion Roadmap

Goal: Build a supportive desktop/web application (“Clinical Lab Companion”) to help 4th-year MedTech students in the Philippines review protocols, reference ranges, and interpret practice lab results during their clinical internships and special topics courses.

📅 8-Week Timeline Overview

Week	Focus Area	Deliverable / Milestone
1	Python & Environment Setup	Install Python, IDE, create virtual environment, learn basic syntax (variables, loops, functions).
2	Streamlit & Project Scaffold	Install Streamlit, scaffold app structure, create a homepage mockup with sidebar navigation.
3	Data Collection & Storage	Build JSON/CSV files for tube protocols and CBC reference ranges. Load and display in the app.
4	Protocol Lookup Feature	Implement tube color & inversion lookup with search bar. UI displays details dynamically.
5	Reference Range Feature	Implement lab value input form, highlight high/low values against JSON/CSV ranges.
6	Interpretation Helper (Study Mode)	Add logic to flag abnormal values and show study hints (e.g., “Consider anemia patterns”).
7	Lecture Summarizer MVP	Integrate simple text-upload area; extract key headings to generate bullet-point summaries.
8	Testing, Polish & Deployment	QA with MedTech friends, fix bugs, add local deployment instructions, package app.

🔨 Weekly Details

Week 1: Python Fundamentals

- **Learning Tasks:**
- Install Python 3.10+ and VS Code (or PyCharm).
- Create a virtual environment (`python -m venv venv`).
- Master basics: variables, lists/dicts, conditionals, loops, functions.
- Resources: FreeCodeCamp Python tutorial; W3Schools Python reference.
- **Milestone:** Able to write and run simple scripts, e.g., reading a CSV and printing values.

Week 2: Streamlit & Project Scaffold

- **Learning Tasks:**
- Install Streamlit (`pip install streamlit`).
- Create `app.py`: import Streamlit, add title and sidebar placeholders.
- Sketch UI: sections for Protocol Lookup, Reference Ranges, Interpretation, Summaries.
- Resources: Streamlit docs, sample tutorial app.
- **Milestone:** `streamlit run app.py` shows homepage with navigation menu.

Week 3: Data Collection & Storage

- **Tasks:**
- **Collect Data**: Ask friends for their school's CBC reference ranges and BD tube protocol PDFs.
- **Format Data**: Create two JSON files:
  - `tube_protocols.json`: each entry with `tube_color`, `additive`, `inversion_count`, `purpose`.
  - `cbc_ranges.json`: each entry with `test_name`, `min_value`, `max_value`, `unit`, and `critical_low/_high`.
- **Load Data** in Python: use `json` or `pandas` to read and print sample.
- **Milestone:** Data files present and loaded successfully in the app (display raw JSON content).

Week 4: Protocol Lookup Feature

- **Tasks:**
- Build a Streamlit text input or selectbox for tube colors.
- On selection, display additive, inversion count, and purpose from `tube_protocols.json`.
- Style with `st.table` or `st.markdown` for clarity.
- **Milestone:** Interactive Protocol Lookup UI working.

Week 5: Reference Range Feature

- **Tasks:**
- Create input fields for key CBC values (RBC, Hgb, Hct, WBC).
- On form submission, compare each input to the loaded ranges.
- Highlight values out of range (e.g., red text for abnormal).
- **Milestone:** Users can input values and see high/low alerts against their school's ranges.

Week 6: Interpretation Helper (Study Mode)

- **Tasks:**
- Define simple rules: e.g., if Hgb < min → "Possible anemia: consider iron studies."
- Store hints/rationales in JSON alongside ranges.

• Display hint messages below the results table.
• **Milestone:** Abnormal values generate context-sensitive study hints.

Week 7: Lecture Summarizer MVP

• **Tasks:**
• Add file uploader (`st.file_uploader`) for .txt or .pdf notes (start with .txt for simplicity).
• Parse text, split on headings (e.g., lines ending with `:`), extract bullet points under each.
• Display collapsible sections (`st.expander`) with extracted summaries.
• **Milestone:** Users can upload notes and see concise bullet summaries.

Week 8: Testing, Polish & Deployment

• **Tasks:**
• Conduct user testing sessions with MedTech friends during internship downtime.
• Fix UI quirks, data errors, and add instructions.
• Create `requirements.txt` and deployment guide (local: `streamlit run app.py` & optionally `pip install --user`).
• Package into GitHub repo with README and sample data.
• **Milestone:** Fully functional Clinical Lab Companion ready for beta use in real internships.

Personal Learning Roadmap (No Background Required)

Goal: Build up your skills in programming, basic math, and AI technologies alongside the Clinical Lab Companion project. This 8-week roadmap runs in parallel with your app development and helps you gain the necessary foundation to tackle each milestone confidently.

📆 8-Week Learning Timeline

Week	Topics	Outcome / Skills Gained
1	**Python Fundamentals I**	Variables, data types, lists, dicts, basic I/O.
2	**Python Fundamentals II & Git Basics**	Functions, loops, conditionals, simple scripts; Git init, commit, push.

Week	Topics	Outcome / Skills Gained
3	**Data Structures & File Handling**	JSON, CSV, basic Pandas; reading/writing files.
4	**Basic Math for Data & Lab Values**	Algebra essentials, basic statistics (mean, median, ranges).
5	**Intro to AI/ML Concepts & NumPy/ Pandas**	Understanding datasets; NumPy arrays; Pandas DataFrames.
6	**Streamlit & Visualization Basics**	Building simple UIs; plotting with Matplotlib; streamlit widgets.
7	**APIs & Intro to AI Libraries**	REST API concepts; basic usage of OpenAI/GPT or local models.
8	**Deployment & Version Control Advanced**	Publishing on Heroku; Git branching; Docker basics.

🔍Week-by-Week Breakdown

Week 1: Python Fundamentals I

- **Learn:**
- Install Python 3.10+, set up VS Code or VSCode.
- Syntax: variables, strings, numbers.
- Lists & dictionaries: create, access, modify.
- Basic input/output (`print`, `input`).
- **Practice:** Write a script to calculate BMI from user input.

Week 2: Python Fundamentals II & Git Basics

- **Learn:**
- Control flow: `if` / `elif` / `else`, loops (`for`, `while`).
- Functions: definition, parameters, return values.
- Git: install Git, `git init`, `git add`, `git commit`, `git push` to GitHub.
- **Practice:** Build a number-guessing game; version it with Git.

Week 3: Data Structures & File Handling

- **Learn:**
- Work with JSON & CSV using Python standard library and `csv` module.
- Introduction to Pandas: read CSV, basic DataFrame operations.
- **Practice:** Load a CSV of sample lab values, compute averages.

Week 4: Basic Math for Data & Lab Values

- **Learn:**

- Algebra: solving for x, manipulating equations.
- Basic statistics: mean, median, mode, standard deviation.
- **Practice:** Compute mean and standard deviation for sample CBC data in Python.

Week 5: Intro to AI/ML Concepts & NumPy/Pandas

- **Learn:**
- Core ML concepts: supervised vs. unsupervised learning.
- NumPy: arrays, vectorized operations.
- Pandas: filtering, grouping, summary stats.
- **Practice:** Use a small dataset to train a simple linear regression (with scikit-learn).

Week 6: Streamlit & Visualization Basics

- **Learn:**
- Streamlit: basic app setup, widgets (`st.button`, `st.slider`, `st.text_input`).
- Matplotlib: basic plotting (`plt.plot`, histograms).
- **Practice:** Build a mini app to visualize lab value distributions.

Week 7: APIs & Intro to AI Libraries

- **Learn:**
- REST API principles; using Python's `requests` library.
- Intro to AI libraries: scikit-learn workflows; basic TensorFlow or PyTorch setup.
- Optional: OpenAI GPT API for text summarization (free trial keys).
- **Practice:** Fetch data from a public REST API; use GPT API to summarize a medical article.

Week 8: Deployment & Version Control Advanced

- **Learn:**
- Deploying a Streamlit app to Heroku or Streamlit Cloud.
- Git branching strategies: `feature`, `main`, `dev` branches.
- Docker basics: writing a `Dockerfile` for your app.
- **Practice:** Deploy your Clinical Lab Companion MVP; collaborate on GitHub with branches and pull requests.

Tips for Success: - Spend at least 5–7 hours per week on learning + practice. - Pair programming: study with a friend or online partner. - Leverage online tutorials (YouTube, FreeCodeCamp, Kaggle). - Keep a learning journal: log challenges and solutions.

You'll build a rock-solid foundation alongside your project — zero background needed! Let me know if you want more detail on any week. 🚀 **Post-Roadmap Next Steps** 1. **Collect feedback & iterate**: Add more tests (urinalysis, chem panels).
2. **Consider lightweight AI**: Summaries using GPT API, dynamic question generation.
3. **Explore deployment**: Host on Heroku or local network for easy access.

You’re set to start! Each week’s tasks build on the previous, keeping the project manageable and directly useful. Good luck, and feel free to ask for help on any of these milestones! 🚛