**BenchMate Tool Development Overview (Developer Introductory Document)**

**Purpose**

This document provides an overview of the BenchMate scientific platform’s **Benchtop module**, which focuses on the scientific tool suite.  
It defines the system’s architecture, development principles, folder structure, and the strategy for integrating data analysis and visualization tools.

It is intended to onboard developers into a consistent, scalable environment that will support hundreds of scientific techniques, tools, and user-driven workflows over time.

**Big Picture Summary for Developers**

BenchMate Benchtop is an **AI-driven, modular scientific platform** where users can upload their own experimental data, conduct analyses, and generate customizable, interactive visualizations — all without needing programming skills.

As the platform expands, we will add new scientific tools (e.g., Volcano Plot, UMAP, PCA) under well-defined Domains (e.g., Transcriptomics, Proteomics) and Techniques (e.g., Bulk RNA-seq, Mass Spectrometry).

Each tool will follow a **common development pattern** on both backend and frontend to ensure ease of scaling, maintenance, and user experience.

**Core Development Principles**

* **Absolute Imports:**  
  All code should use absolute imports to simplify module referencing and improve maintainability.
* **Dynamic Routing:**  
  Frontend routes will be generated dynamically from a configuration array, enabling dynamic loading of different tools and reusing common layouts where possible.
* **Config-Driven Tools:**  
  Each tool will have its own YAML configuration file, defining:
  + Expected user input
  + Default thresholds and visualization settings
  + Backend processing logic
  + Frontend customization options
* **Backend-Frontend Contract:**  
  Each tool will define expected input/output formats.  
  Backend and frontend must communicate through clear, structured JSON contracts, allowing flexibility in user data while maintaining stability in processing.
* **User Data Variability Handling:**  
  Frontend must allow users to **map their own columns** (e.g., which column is p-value, which is log2FC) to accommodate variability in user-uploaded data.
* **Common Layouts and Components:**  
  Reusable frontend components (e.g., UploadForm, SettingsSidebar, PlotCanvas) will be used wherever possible to maintain a unified user experience.
* **Statistical Analysis and Interaction:**  
  Where relevant, tools will offer statistical testing and dynamic data interaction features.  
  Users must be able to manipulate thresholds, filters, and download their customized datasets.
* **Security and Privacy:**  
  Backend will plan for secure handling of user-uploaded files. Such as data that only persists in that session state unless they choose to save it or publish/post it  
  Future authentication layers are considered as the platform grows.
* **Versioning and Testing:**  
  APIs and components will be designed with forward-compatibility and testing in mind to allow easy future updates.

**Future Vision (Already Considered in Design)**

* Integration of an **AI Assistant** for users to interact conversationally with their data.  
  The AI will only analyze user-uploaded data to avoid hallucinations and ensure trustworthy assistance.
* Expansion to **interdisciplinary collaborations** by connecting users with similar datasets, problems, techniques, or observations, strengthening the BenchMate community.
* Future scaling to incorporate secure authentication, multi-user projects, and collaborative data workflows.