Application Instructions

**Technology Requirements**

* The codebase **must be written entirely in Python**, using suitable libraries such as:
  + **Streamlit**, **Dash**, or **Tkinter** for the interface
  + **Plotly**, **Matplotlib**, or **Seaborn** for charting
  + **Pandas** and **NumPy** for data handling
* The application must be **modular** and cleanly organised into separate files and folders:

/project\_root/

main.py # Application entry point

/components/ # UI elements (charts, sliders, forms)

/services/ # Business logic, projections, calculations

/data/ # Static or reference datasets (e.g., CSVs)

/utils/ # Helper functions and constants

/database/ # Database models and connection logic

/assets/ # Static files (images, icons, CSS)

config.py # Configuration settings (DB URL, toggles, etc.)

**Database Integration**

* All user data (investment inputs, portfolio configurations, and selected ETFs) must be **stored in a database**.
* Support common lightweight or cloud databases such as **SQLite**, **PostgreSQL**, or **MongoDB**.
* Implement database operations using **ORM tools** like **SQLAlchemy** or native connectors.
* The code must be designed so that the database backend can be **easily swapped or upgraded** (e.g., by changing config.py or environment variables).
* Include basic **CRUD operations** for portfolios and ETF selections.

**Page 1: User Setup**

* Input fields:
  + Full name and surname
  + Initial investment amount (ZAR)
* Interactive elements:
  + Portfolio allocation slider (Tech vs Complementary ETFs)
  + Investment duration selector (1–10 years)
  + Optional: risk tolerance selection (Low / Medium / High)
* Field validation with tooltips
* Store input into the database
* Redirect to Portfolio Dashboard upon confirmation

**Page 2: Portfolio Dashboard**

**Interactive Visuals:**

* **Pie chart** for portfolio allocation
* **Line/bar chart** for projected growth over time
* **S&P 500 benchmark comparison** line
* Optional toggles for **monthly, quarterly, yearly** projections

**Real-Time Interactivity:**

* Any updates to investment amount, contributions, duration, or allocations should **instantly reflect in all calculations and charts**

**User Controls:**

* Adjust:
  + Initial investment
  + Monthly/yearly contributions
  + ETF allocations (via sliders or numeric inputs)
* Add/remove ETFs from a preloaded list of top 30 (15 Tech, 15 Complementary)

**Data & Projections:**

* Use **historical ETF data** (loaded from /data/ or from the database)
* Simulate projections using:
  + Historical CAGR or average annual return
  + Optional projection mode (Conservative / Average / Aggressive)
* Show:
  + Total projected value
  + Year-on-year portfolio performance
  + Average annual return
  + Cumulative profit
  + Visual risk indicator
  + Alpha performance vs. S&P 500 (e.g. “+1.2% vs S&P”)

**Dashboard Summary (Always Visible)**

* Current investment breakdown
* Final projected value
* Total profit
* Contributions
* Risk level
* ETF list with allocation and return stats
* Performance vs. S&P

**Additional Features**

* **Download report** as PDF or CSV (summary and graphs)
* **Light/Dark mode** toggle
* **Responsive design** (mobile/tablet friendly)
* **Data persistence** using the connected database
* **Session saving/loading** for users
* Easily exportable and deployable app with database reconfigurability

**Developer Notes**

* Ensure all file paths and dependencies are relative and environment-agnostic
* Keep config values (e.g. database URL, debug mode) in a single config.py or .env file
* Provide clear documentation or a README.md on how to set up, run, and switch databases