Comprehensive Analysis and Dietary Strategies with Tableau: A College Food Choices Case Study

To implement this case study, a blend of data visualization, web technologies, and data processing tools were used. The stack was selected to ensure scalability, ease of use, and performance.

---

1. Data Visualization Layer

Tool: Tableau Desktop / Tableau Public
Purpose:

- Create interactive dashboards and story scenes
- Visualize survey results, nutrient intake, spending patterns
- Provide persona-based segmentation insights

Features Used:

- Bar charts, pie charts, treemaps, heatmaps
- Dashboard filters and story points
- Calculated fields, blending multiple data sources

---

2. Data Processing & Preparation

Tools:

- Microsoft Excel / Google Sheets
- Python (with Pandas, NumPy, Matplotlib for preprocessing)

Purpose:

- Clean and preprocess raw student food logs, survey responses
- Perform calculations (e.g., BMI, calorie intake, budget balance)
- Export structured datasets to Tableau-compatible formats (.csv/.xlsx)

---

3. Web Integration Layer (Optional for Interactive Demo)

Framework: Flask (Python Microframework)

Purpose:

- Host the dashboards and results in a web-based environment
- Provide a simple web interface for uploading student food logs or survey data
- Embed Tableau dashboards using iframe or Tableau JS API

Key Libraries:

- `Flask`, `Jinja2`, `Werkzeug`
- `Requests` (for pulling external API data if needed)

---

4. Backend & Data Storage (Simple Setup)

Option A (Offline):

- Flat file storage (`.csv`, `.xlsx` files)

Option B (Online/Dynamic):

- MongoDB Atlas (NoSQL, for survey responses and logs)
- PostgreSQL or SQLite (if relational DB is preferred)

---

5. Deployment & Hosting

Local Testing:

- Flask app run locally using `localhost:5000`

Hosting Options:

- GitHub Pages (for static dashboards)
- Heroku / Render (for Flask web app)
- Tableau Public for publicly sharing dashboards

---

6. Security & Access (Optional if Extended)

Features (if login or data privacy is implemented):

- User authentication via Flask-Login
- Input sanitization and form validation
- Dashboard access controls (e.g., user role-based)

Stack Overview Table

Layer	Tool/Tech Used	Purpose
Visualization	Tableau	Dashboard, charts, trends
Data Prep	Python (Pandas), Excel	Cleaning, calculations, data formatting
Web Integration	Flask	Serve app, embed Tableau
Storage	CSV, MongoDB/PostgreSQL	Store logs and responses
Hosting	Tableau Public, Heroku, GitHub	Share dashboards, deploy app
Optional Security	Flask-Login, WTForms	Protect user data, input validation