Week 1: Project Initiation

* Gather project requirements and define scope.
* Create project plan and timeline.
* Set up development environment.

Week 2-3: User Account Management

* Develop user registration functionality.
* Design user registration form using Flask-WTF.
* Implement validation for registration inputs.
* Store user account information in a Flask-SQLAlchemy database.
* Implement secure user login functionality.
* Design login form using Flask-WTF.
* Verify user credentials and authentication using Flask-Login.
* Set up session management with Flask-Session.

Week 4-5: Photo Upload and Comparison

* Develop photo upload functionality.
* Design photo upload interface using Flask-WTF.
* Implement file upload and storage with Flask-Uploads or Flask-S3.
* Implement photo comparison and visualization.
* Design comparison algorithms using NumPy and OpenCV.
* Develop UI for comparing photos with Flask templates and Bootstrap.

Week 6-7: Metrics Tracking

* Develop input forms for weight and macro metrics using Flask-WTF.
* Design UI for entering metric values.
* Implement data validation for metric inputs.
* Develop functionality to store and track metrics using Flask-SQLAlchemy.
* Create database schema for metric storage.
* Implement data storage and retrieval with Flask models.

Week 8-9: Graphical Visualization

* Create graphs and visual representations of tracked metrics using Flask and libraries like Matplotlib or Plotly.
* Choose suitable graph types (line, bar, etc.).
* Generate graphs dynamically based on stored metric data.
* Implement dynamic graph functionality with Flask and JavaScript.
* Enable filtering and customization of graphs.
* Implement interactive features (hover, click, etc.) using JavaScript libraries.

Week 10-11: User Roles and Permissions

* Implement user roles and access control with Flask-User or Flask-Security.
* Design user role management interface.
* Assign roles to users based on their authorization.
* Develop coach-client authorization mechanism with Flask.
* Design authorization workflows specific to your application.
* Implement access restrictions for client pages.

Week 12-13: Performance Optimization and Testing

* Conduct performance testing and optimization using Flask profiling tools and performance analysis.
* Identify performance bottlenecks.
* Optimize database queries and server-side code.
* Conduct comprehensive testing of all features.
* Develop test cases and scenarios.
* Perform functional and regression testing with Flask testing tools.

Week 14-15: Finalize and Polish

* Fine-tune user interface and user experience with Flask templates, CSS, and JavaScript.
* Refine UI layout and design.
* Enhance usability and responsiveness.
* Address any remaining bugs or issues.
* Collect and prioritize bug reports.
* Debug and resolve identified issues.

Week 16: Documentation and Deployment

* Prepare comprehensive documentation.
* Create user manuals and guides.
* Document system architecture and APIs using tools like Sphinx.
* Deploy the software to a production environment using Flask deployment strategies.
* Set up production servers (e.g., Nginx, Gunicorn, or uWSGI).
* Configure and optimize deployment settings.
* Perform final testing and readiness checks.