Project Setup:

Create a new directory for your project

Set up a virtual environment

Install necessary Python packages (Flask, firebase-admin, flask-login, etc.)

Create a .gitignore file

Firebase Setup:

Create a new Firebase project in the Firebase Console

Enable Google Sign-In method in Firebase Authentication

Download the Firebase Admin SDK private key (JSON file)

Add localhost and 127.0.0.1 to authorized domains in Firebase Console

Flask Application Structure:

Create app.py (main application file)

Create config.py (configuration settings)

Create auth.py (authentication routes and logic)

Create models.py (database models)

Create extensions.py (Flask extensions)

Create a templates folder with index.html and dashboard.html

Environment Variables:

Create a .env file to store sensitive information

Add Firebase configuration details to .env

Firebase Integration:

Initialize Firebase Admin SDK in app.py

Set up Firebase configuration in index.html

Flask Application Setup:

Set up basic Flask application in app.py

Configure the application (secret key, database URI, etc.)

Initialize Flask extensions (SQLAlchemy, LoginManager)

Database Models:

Create User model in models.py

Authentication Logic:

Implement login route in auth.py

Implement logout route in auth.py

Create Firebase token verification function

Frontend Setup:

Design login page (index.html)

Design dashboard page (dashboard.html)

Add Firebase SDK and authentication logic to index.html

Route Protection:

Implement login\_required decorator for protected routes

User Session Management:

Implement user loader for Flask-Login

Error Handling:

Implement error handlers for common HTTP errors

Testing:

Test the login flow

Test protected routes

Test logout functionality

Deployment Preparation:

Update configuration for production environment

Ensure all sensitive data is in environment variables

Documentation:

Write README.md with setup instructions

Document any complex parts of your code

Version Control:

Initialize Git repository

Make initial commit

Create .gitignore file to exclude sensitive files

Additional Features (if time permits):

Implement user profile page

Add password reset functionality

Implement email verification

Security Checks:

Ensure all routes are properly protected

Verify CSRF protection is in place

Check for any exposed sensitive information

Performance Optimization:

Optimize database queries

Implement caching if necessary

Final Testing:

Perform end-to-end testing of the entire application

Test on different browsers and devices