Project Plan

Stages:

1. Coordination and Standardization:
   * Data Collection Protocols: Make sure all students follow agreed-upon protocols for data collection, cleaning, and formatting. This ensures consistency across all datasets.
   * Standard API Specifications: Since each group will provide their cleaned data via an API, agreeing on a standard API specification (e.g., RESTful principles, response formats) is crucial. This will make it easier to integrate data from different groups.
   * Documentation and Metadata: Ensure each dataset is accompanied by comprehensive documentation and metadata. This should include details about the source, the nature of the data, any transformations applied, and how to access it through the API.
2. Central Integration Point:
   * Central API Gateway: Implement a central API gateway that aggregates all individual APIs. This central point will simplify access for users who need data from multiple sources and provide a single point of entry for data queries.
   * Authentication and Authorization: Consider implementing authentication and authorization mechanisms at this gateway to control access and ensure data security.
3. Data Quality and Validation:
   * Validation Mechanisms: Set up mechanisms to validate the data as it comes in from different sources. This can include checks for data completeness, accuracy, and adherence to the SDMX standards.
4. Data Accessibility:
   * Design the central data access points to be user-friendly, providing easy-to-use interfaces or query capabilities that cater to a variety of users, from technical staff to decision-makers.

Consider:

* System Monitoring: Implement monitoring tools to track the health of the APIs and the central system.
* Design the system architecture with scalability in mind, allowing for easy addition of new data sources or expansion of the user base without significant redesign.
* Conduct load testing to understand how the system performs under stress and to identify potential bottlenecks.

**Architecture Overview**

**a. Data Ingestion**

* Data Collection: Students/groups upload their cleaned data in JSON format to an S3 bucket. This can be set up to trigger a Lambda function.
* Use Lambda for initial processing or transformation of this data as needed. This is especially useful for light transformations before loading into your data warehouse.

**b. Data Storage and Processing**

* Database/Redshift: Store processed data in Amazon RDS for operational queries or Amazon Redshift for analytical queries.
* Multi-AZ Deployments for RDS/Redshift: Ensure high availability and fault tolerance by deploying your databases across multiple availability zones.

**c. API Management**

* API Gateway: Setup API Gateway to handle requests to and from your data storage. It serves as the front door for all API requests to access data, enforcing throttling rules and handling API version management.

**d. Authentication and Authorization**

* AWS Cognito: Manage user authentication and provide controlled access to the API, ensuring that only authenticated users can make requests.

**e. Frontend Web Application**

* Elastic Beanstalk/EC2: Deploy your web application on Elastic Beanstalk for easy scaling and management, or directly on EC2 instances if you require more control over the environment.
* CloudFront: Use CloudFront to distribute the web application content globally, reducing latency and improving load times for end-users.