Various approaches for the project along with their pros & cons

The various approaches to solve for this were based on different architectural styles which are:

1) Monolithic Architecture
2) Serverless Architecture
3) Microservices Architecture
4) Event-Driven Architecture
5) Layered Architecture

As the problem statement specified that the project should be designed in plug-and-play fashion which should also be scalable, the optimal architecture that I could think of from the above was **Microservices based architecture**. Since the challenge consisted of creating a data collection platform along with post-submission business logic, each of these could be considered as a micro service to the main data center which could make use of the platform with the exact plug-n-play fashion. There is a main server that communicates with the database and for each of the microservice there exists their own independent server which can communicate with the main server via APIs. In this way it can also be scaled to a large extent as well.

Pro/Con Analysis

A. Monolithic Architecture

a. Pros:
   i. Simplicity as it is easier to develop, test and deploy.
   ii. Single Codebase due to which maintenance is relatively straightforward as everything is in one place.
   iii. Performance is good as there is lower overhead due to direct method calls.

b. Cons:
   i. Scalability is an issue.
   ii. Not flexible due to limited technology stack for different components.
   iii. Fault Isolation cannot be prevented as issues in one module can affect the entire system.

B. Serverless Architecture

a. Pros:

i. It offers something called automatic scaling which is handled by cloud providers.
ii. This does not contain any fixed infrastructure costs so it requires payment only for actual usage.
iii. This architecture promotes developers to focus on writing code without managing servers.

b. Cons:
i. Unfortunately it has initial latency when functions are triggered which is also known as Cold Starts.
ii. Another drawback is that it gets tied to a specific cloud provider's serverless offering.
iii. Since it is serverless, the serverless platform imposes time limits to functions.

C. Microservice Architecture

a. Pros:
i. The best part of this architecture is that independent services can be scaled individually.
ii. It is very flexible due to which technology stack can vary between services.
iii. It promotes CI/CD and is also fault isolated as issues in one service do not necessarily affect the other.

b. Cons:
i. However in terms of complexity, there is an increase in it in terms of service communication and data consistency.
ii. In order to monitor and manage the distribution of the system, additional tools are required.
iii. Due to inter-service communication, latency gets introduced.

D. Event-Driven Architecture

a. Pros:
i. This architecture promotes decoupling as components communicate through events.
ii. It is very well suited for real-time processing.

iii. This architecture too is very feasible for scaling horizontally.

b. Cons:
   i. Once again, such architecture is complex to design and maintain.
   ii. Even though it is implemented, debugging and tracing events is challenging.
   iii. Also in order to ensure consistency across event-driven components, extra effort is required.

E. Layered Architecture

a. Pros:
   i. As the name suggests it clearly defines layers for presentation, business logic and data access.
   ii. Hence it gets easier to maintain and extend.
   iii. It also allows for the replacement of individual layers without affecting others, therefore modularity is maintained.

b. Cons:
   i. One of the first drawbacks of this architecture is that it has high rigidity as changes in one layer impact others.
   ii. Additionally, in order to scale, it might require to scale the entire layer which too gets hefty.
   iii. Unfortunately for each of its layers, there is less flexibility in terms of technology choices.