Azure Functions Chaining

CST8917 Serverless Applications

May 23rd, 2025

What is Azure Functions?

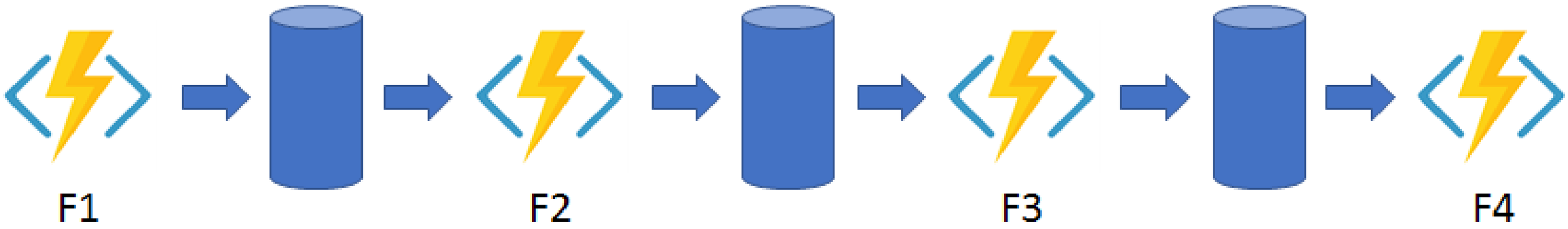
- A serverless compute service that enables you to run **event-driven** code without managing infrastructure.
- Great for microservices, event-based processing, and background tasks.

The Challenge: Stateless Execution

- Functions are inherently **stateless and short-lived**.
- Complex workflows need **state management, retries,** and **chaining**.
    - **Example:** Data processing pipelines.

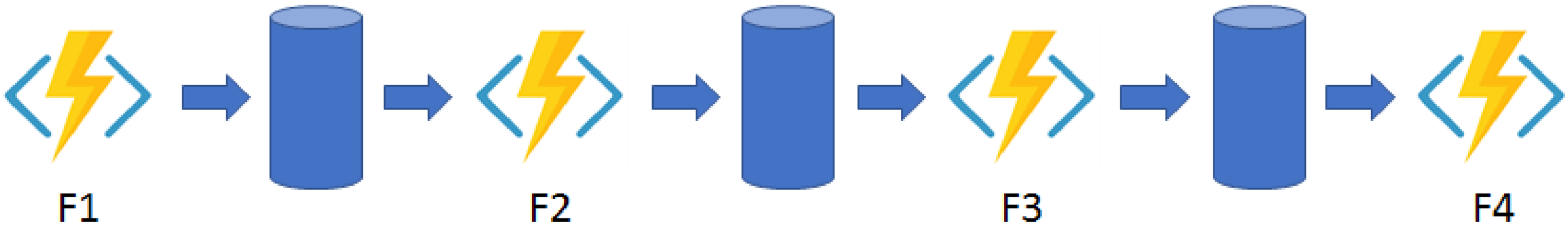
Serverless Function Chaining

- A **pattern** used in serverless architectures where multiple serverless functions are linked together to perform **complex or multi-step** tasks.



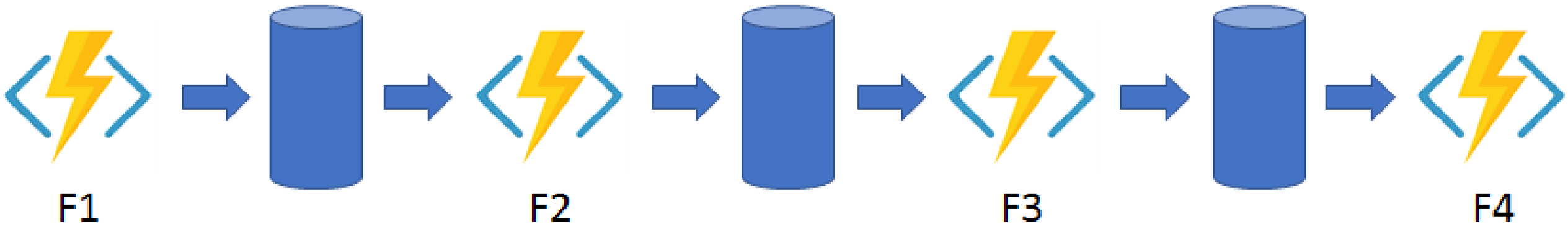
Serverless Function Chaining

- This pattern is useful because serverless functions are typically designed to be **small, single-purpose, and stateless**.
- By chaining them together, you can create **more sophisticated** and capable applications.



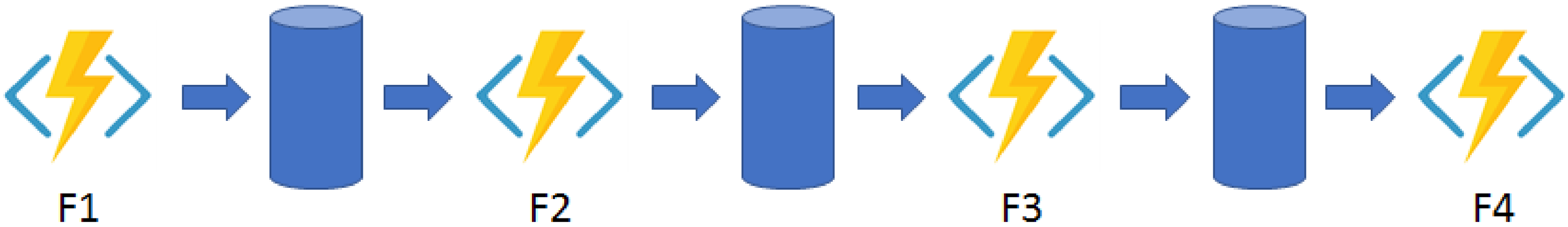
Serverless Function Chaining

- Function Chaining involves invoking one function after another, in a **sequence**.
- Each function in the chain performs its task and then **triggers** the next function in the sequence, passing along any necessary data.
- The **output of one function** can be the **input to the next**, creating a chain of operations.



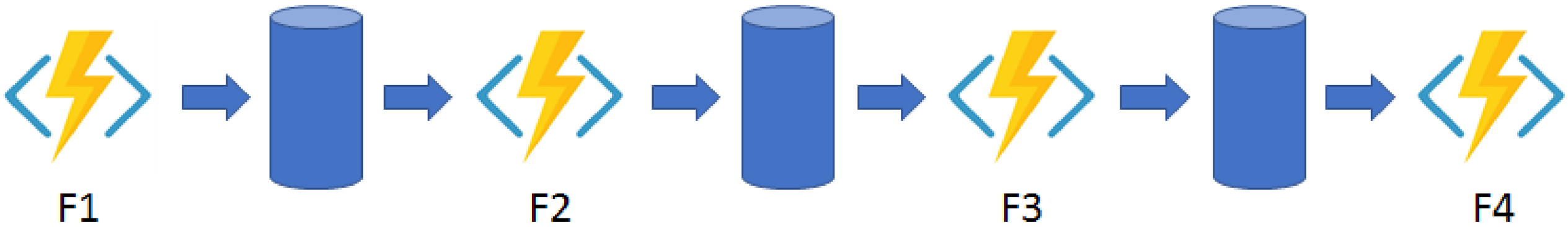
Function Chaining: How it works?

- **Trigger:** The chain can be initiated by an event or a manual trigger. This event invokes the first function in the chain.



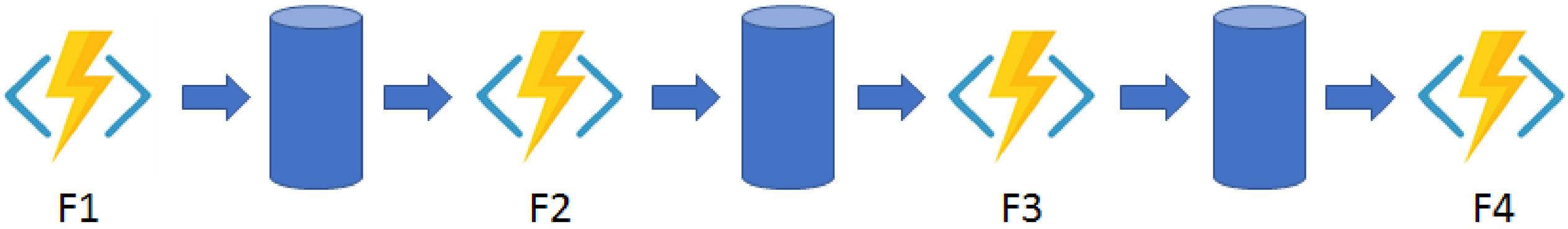
Function Chaining: How it works?

- **Execution:** The first function processes the event, performs its operation, and then invokes the second function, possibly passing along data.



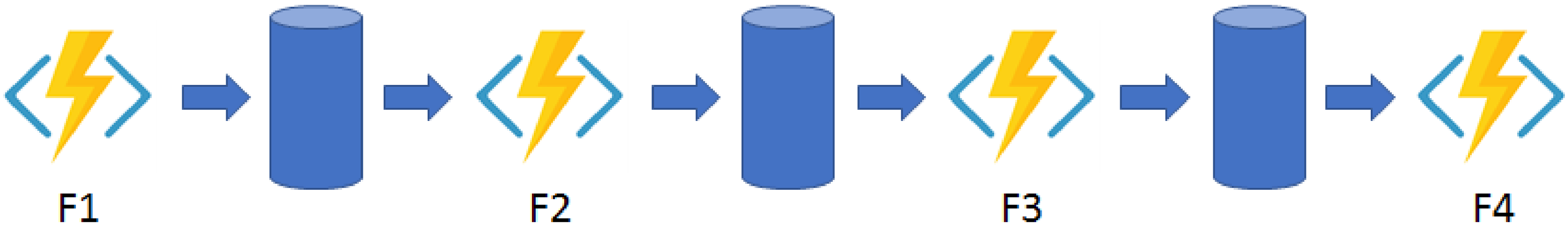
Function Chaining: How it works?

- **Data Passing:** Functions can pass data from one to the next. This can be done through direct invocation or by using intermediary storage (like a database or cloud storage).



Function Chaining: How it works?

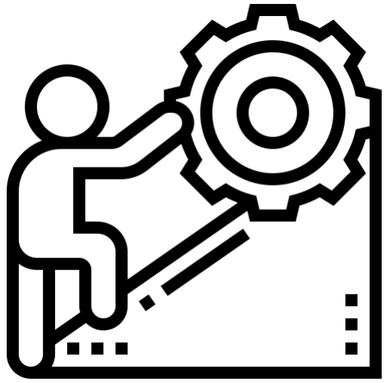
- **Completion:** The chain continues until the final function executes. The final output can be stored, returned to the user, or trigger another process.



Advantages

- **Scalability:** Each function can independently scale.
- **Maintenance:** Smaller, individual functions are generally easier to manage and update.
- **Cost-Effectiveness:** You pay only for the execution time of each function, which can be more cost-effective than running full servers.

Challenges



- **Complexity in Monitoring and Debugging:** Tracking the flow across multiple functions can be challenging.
- **Cold Starts:** If a function hasn't been used recently, it may take longer to start, which can impact performance.
- **Orchestration:** Managing the sequence and handling errors or retries can add complexity.

Orchestration Tools

To manage the complexities of function chaining, cloud providers offer orchestration services (like **AWS Step Functions**, **Azure Durable Functions**, or **Google Cloud Workflows**) that manage the function workflow, error handling, and state management.

Azure Function Chaining

- A **specific implementation** of the broader concept of Serverless Function Chaining within the Microsoft Azure cloud platform.
- It utilizes **Azure Functions**, which are Microsoft's offering in the Function as a Service (FaaS) market, to build scalable and event-driven applications.
- Azure provides multiple ways to implement function chaining, but a common and powerful method is by using **Azure Durable Functions**.

Azure Durable Functions

- An **extension** of Azure Functions that lets you write **stateful** functions in a serverless environment.
- It provides powerful capabilities to **orchestrate function execution** in a **sequence** or in **parallel**, manage state, and handle long-running and complex workflows.

Azure Durable Functions: Advantages

- **Built-in State Management:** Azure Durable Functions manage state, checkpoints, and restarts for you, simplifying the process of chaining functions.
- **Orchestration:** It provides a rich set of orchestration capabilities, allowing you to define complex workflows as code.
- **Integration:** Seamless integration with other Azure services and resources.

Orchestrator Functions

- Orchestrator functions **define workflows** by using normal procedural code.
- The functions can call other functions, and the state of the workflow is maintained by **Durable Functions**.

Activity Functions

- The **basic unit of work** in a Durable Functions application.
- They're the functions that do the actual work of your application.
- Orchestrator functions **call one or more activity functions**.

Client Functions

- A **regular Azure Function** that starts an orchestrator function.
- Example: HTTP triggered function.

Home Exercise – Cold Starts in Serverless Computing

- Assigned Reading: Durable Functions Overview - Azure | Microsoft Learn
- **Task:**
  - Read the article carefully.
  - **Make sure that you can answer the followings questions:**
    - What is Durable Function?
    - What typical application patterns that can benefit from Durable Functions?