ESA Assignment-1   
Architecture for an online Bill Payment solution.

Name: Shagufta Anjum  
Roll no: 19XJ1A0568

The solution contains the following major components:

* Stakeholders – Customers, Utility companies
* High level design using layered design
* View past transactions/upcoming bills
* Bill payment using event-driven mediator design
* Payment Gateway using broker pattern
* Notification service (Email and SMS) using pipes and filters
* Distributed database

A break-down of each of the major components involved:

1. First, a high-level view - the entire solution is designed with a layered architecture having 4 layers

Diagram

Description automatically generated

Breaking each component down further,

1. Pay Bill component using an event-based design with a mediator topology

Diagram

Description automatically generated

1. Payment Gateway using a broker architecture

Diagram

Description automatically generated

1. Notification system (email and SMS) using pipes and filters

Diagram

Description automatically generated

1. Viewing bills and transaction data from app using request and response

Diagram

Description automatically generated

**Reasons for choosing these patterns**:

The solution above uses different design patterns for different components to ensure easy implementation and scalability of each component:

* Layered architecture is easy to deploy – during development, employee skillsets can be separated by layers. It is not the easiest to scale, but it can be scaled by splitting the layers into separate physical deployments or replicating the entire application across multiple nodes.
* The bill payment component is implemented using event-driven pattern with mediator topology because it involves a stream of incoming events (bill payment requests) and consists of a series of sub steps that need to be orchestrated. Scalability is ensured due to highly independent and decoupled event processors.
* The payment gateway uses a broker pattern because it allows easy addition and removal of components (customers, companies, and banks). It removes the need for customers to directly communicate with banks. Also, the communication is isolated, meaning that if mode of communication changes, the application doesn’t need to change. To ensure scalability, we can add additional brokers with load balancers.
* The email and SMS notification workers will most likely have different scalability requirements, which makes pipes and filters a good option. Also, certain common task workers can be shared.