From Basic to Durable Design Failure Point Analysis

We had a rather basic configuration with only one web server, one backend server, and a single database in the first version of the system (Part A). Though it looked good on paper, in actual events it carried several risks that could significantly compromise everything.

Part A: What Could Go Wrong:

1. One machine doing everything: Should the only web or backend server fail; the whole service will stop. Users would immediately experience downtime; there is nothing to take over.

2. Every item was in one location (one AZ); therefore, should anything go wrong in that zone a power outage, hardware crash, network failure everything would go down all at once.

3. Only one region contained the RDS MySQL instance; the database was not secure. The database would be unavailable and customers wouldn't be able to order or see product information if it failed or required updates.

Part B's Cure for All:

1. We now run at least two EC2 instances in Auto Scaling Groups for both the web and backend servers. Should one drop, another is already running. Should traffic spike unexpectedly, additional servers may spin up automatically.

2. Two Availability Zones are used for all the servers now. In this manner, the system stays live from the second one even if one zone crashes.

3. Our database is now backed by AWS's Multi-AZ deployment, therefore we shifted to RDS Multi-AZ. Zero downtime and no data loss; a full standby copy in another zone automatically takes over if the main one fails.

Final remarks: Part A, the fundamental system, was suitable for a demonstration but not secure for actual use. The revamped configuration (Part B) is considerably more dependable. Without bothering consumers, it can manage server crashes, traffic peaks, and even AZ outages. We chose the robust model as a result.