AWS Elastic Beanstalk(EB)

- You simply upload your application, and Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.
- Supports Go, Java, .NET, Node.js, PHP, Python, and Ruby.
- It requires no or minimal code changes to go from development machine to the cloud.
- Elastic Beanstalk provides a unified user interface (UI) to monitor and manage the health of your applications.
- Elastic Beanstalk uses Elastic Load Balancing and Auto Scaling to automatically scale your application in and out based on its specific needs.
- You don't need any experience working in cloud for using Elastic Beanstalk
- Deploy and manage your applications within minutes in AWS cloud.
- Interact with Elastic Beanstalk using console, AWS CLI, high-level CLI designed by EB
- Working of Elastic Beanstalk:
  - Create an application (java app)
  - Upload version in form of source bundle (e.g. java .war file)
  - Launch environment (provide some info about the application)
  - Manage environment

Deployment options:

- By default, your environment uses all-at-once deployments.
  - All at once
  - Rolling
  - Rolling with additional batch
  - Immutable
  - Traffic splitting
  - Blue/green deployment
- All at once
  - Quickest deployment method.
  - Application might be unavailable to users for short time, because Elastic Beanstalk deploys new version of app to each instance. So, app server or web server might need to restart.
- Rolling
  - splits environment's EC2 instances into batches & deploys the new version of application to one batch at a time.
  - Avoids downtime, minimizes reduced availability but take longer deployment time.
  - Rest of the environment running old version of the application.
  - Can configure **Batch Size** (percentage or fixed number)
- Rolling with an additional batch
  - Use this deployment if want to maintain full capacity.
  - Avoids downtime, minimizes reduced availability but take even longer deployment time compared to Rolling method.
  - Elastic Beanstalk launches an extra batch of instances, then performs a rolling deployment.
  - same bandwidth is retained throughout the deployment.
  - avoids any reduced availability, although at a cost of an even longer deployment time compared to the Rolling method.
  - Can configure **Batch Size** (percentage or fixed number)
  - If a deployment fails after one or more batches completed successfully, the completed batches run the new version of your application while any pending batches continue to run the old version.
  - If you terminate instances from the failed deployment, Elastic Beanstalk replaces them with instances running the application version from the most recent successful deployment.
- Immutable
  - Slower than other deployment options.
  - launch a full set of new instances running the new version of the application in a separate Auto Scaling group, alongside the instances running the old version.

    - Quick and safe rollback if deployment fails.
    -
- Traffic-Splitting
    - let you perform canary testing as part of your application deployment
    - launches a full set of new instances just like during an immutable deployment. It then forwards a specified percentage of incoming client traffic to the new application version for a specified evaluation period
    - If the new instances stay healthy, Elastic Beanstalk forwards all traffic to them and terminates the old ones. If the new instances don't pass health checks, or if you choose to abort the deployment, Elastic Beanstalk moves traffic back to the old instances and terminates the new ones.
    - There's never any service interruption
    - Can configure Traffic Split & Traffic Splitting evaluation time
- Some policies replace all instances during the deployment or update. This causes all accumulated EC2 burst balances to be lost. It happens in the following cases:
    - Managed platform updates with instance replacement enabled
    - Immutable updates
    - Deployments with immutable updates or traffic splitting enabled
- Blue/Green deployment
    - In this, deploy new version to a separate environment, then swap CNAME to redirect traffic to new version.
-

Using Elastic Beanstalk with Amazon RDS

- Elastic Beanstalk + RDS to set up, operate and scale a relational database.
- Two options:
    - Create a new database in RDS
    - Use existing DB created by elastic beanstalk, then decouple
- To allow the Amazon EC2 instances in your environment to connect to an outside database, configure an additional security group for the Auto Scaling group that's associated with your environment.
- After database instance is launched and security groups configured, the connection information can be passed, such as the endpoint and password, to your application by using environment properties.
- For additional security, store connection info in S3 and retrieve during deployment.
- Externally connected database is not dependent on elastic beanstalk environment, that's why it is not deleted when environment is terminated.
-

Using Elastic Beanstalk with Amazon S3
Using Elastic Beanstalk with Amazon VPC

Advanced environment customization with configuration files

- You can add AWS Elastic Beanstalk configuration files (.ebextensions)
  Example
  .ebextensions/network-load-balancer.config
-