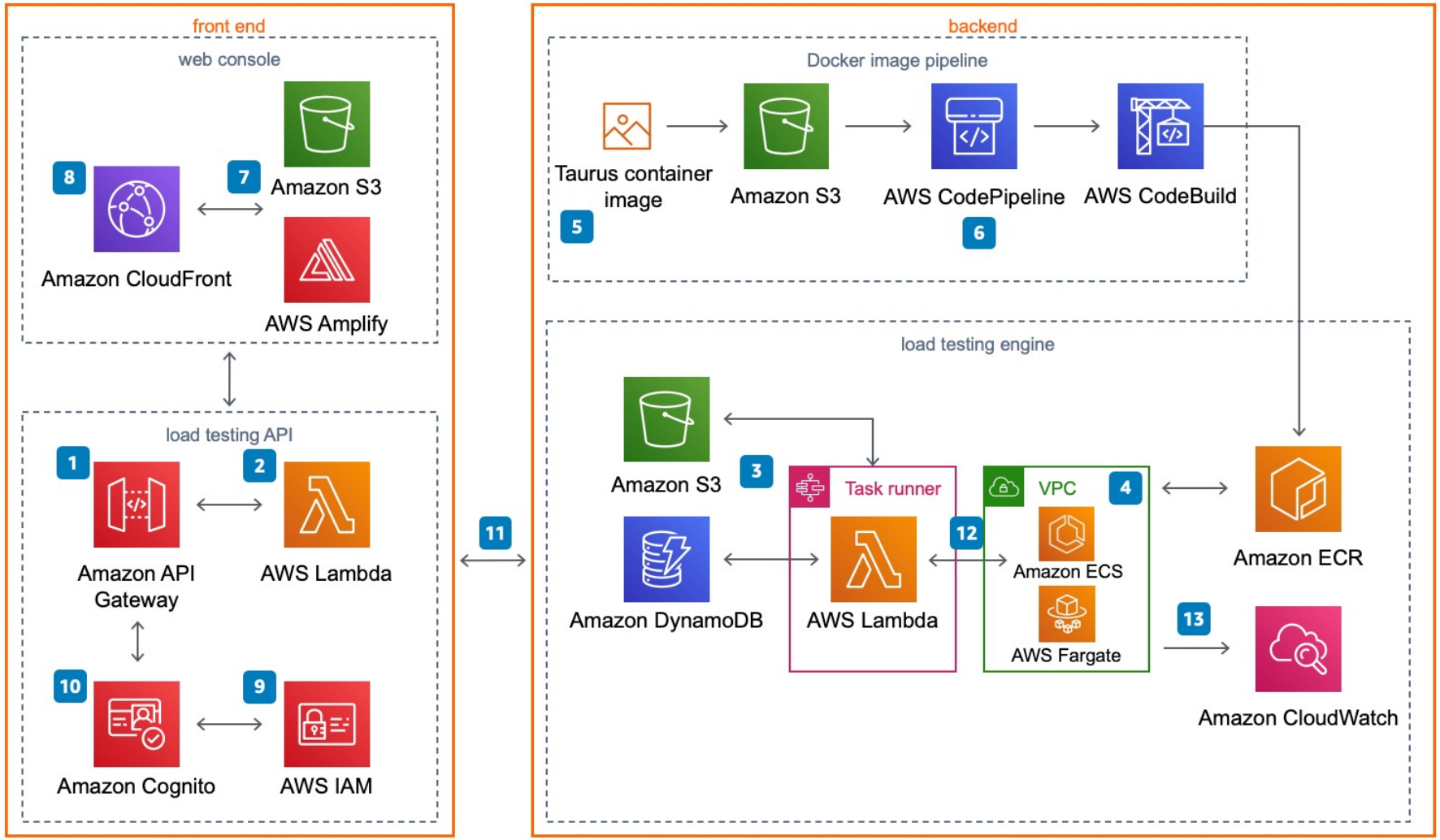
Distributed Load Testing on AWS

This solution helps industrial customers automate the testing of software applications at scale and at load to identify bottlenecks before their release. To deploy this solution using the available AWS CloudFormation template, select **Deploy with AWS**.



1. An **Amazon API Gateway** API invokes the solution's microservices (**AWS Lambda** functions).
2. The microservices provide the business logic to manage test data and run the tests.
3. The microservices interact with **Amazon S3**, **Amazon DynamoDB**, and **AWS Step Functions** to provide storage for the test scenario details and results and run test scenarios.
4. An **Amazon VPC** network topology is deployed containing the solution's **Amazon ECS** containers running on **AWS Fargate**.
5. The containers contain the latest version of the Taurus load testing Open Container Initiative (OCI) compliant container image, which is used to generate load for testing your application's performance.
6. **AWS CodePipeline**, **AWS CodeBuild**, and Amazon S3 help manage the image.
7. A web console powered by **AWS Amplify** is deployed into an Amazon S3 bucket configured for static web hosting.
8. **Amazon CloudFront** provides secure, public access to the solution's website contents.
9. During initial configuration, the solution creates a default administrator role and sends an access invite to a customer-specified email address.
10. An **Amazon Cognito** user pool manages user access to the console and the load tester API.
11. After you deploy this solution, you can use the web console to create a test scenario that defines a series of tasks.
12. The microservices use this test scenario to run Amazon ECS on AWS Fargate tasks.
13. When each task is complete, the results are stored in Amazon S3 and the output is logged in Amazon CloudWatch. Once all tasks are complete, the results are stored in Amazon DynamoDB.

aws **Deployable AWS Reference Implementation**

Deploy with AWS