



**niyantrac**  
Infinite Control. Zero Chaos



Product Installation Guide

Document Version: 1.2

# Getting Started

## 1. Launching the Instance

- Subscribe to the Nyantrac AMI from the AWS Marketplace.
- Launch an EC2 instance using this AMI.
- We recommend using a t3.medium instance or higher for smooth performance.

## 2. Security Group Configuration

- Open Required Ports: Ensure that the following Ports are open in your Instance's Security Group.

**Note: It is recommended to open the below ports only for internal IP Addresses, do not keep ports open for all.**



- **Port 443: For accessing the Web App.**

## 3. IAM Role Attachment

1. Open IAM Console >> Go to Roles >> Click Create Role.
2. Select Trusted Entity:
  - Choose AWS service.
  - Use Case: EC2.
  - Click Next.
3. Permissions:
  - Attach **CloudWatchReadOnlyAccess** policy here.
  - Click Next >> Add a role named **niantrac-role** >> Create role.
4. Add Inline Policy:
  - After creating, open the role >> Go to Permissions tab >> Click Add inline policy.
  - Choose JSON tab >> Paste the policy json on page 03.
  - Click Review policy, name it **niantrac-inline-policy** >> Create policy.
5. Attach Role to EC2 Instance:
  - Go to EC2 console >> Select instance >> Actions > Security > Modify IAM role.
  - Attach the newly created role.

## 4. Open Browser:

**Wait for a few minutes** and navigate to your instance's public IP address or DNS name using a web browser.

### Access Nyantrac:

- If your instance has a public IP, use it as the entry point:

**https://<your-instance-public-ip>**

- If no public IP is present, the application automatically falls back to the private IP:

**https://<your-instance-private-ip>**

### For login, use the initial login credentials:

**Username:** admin

**Password:** AcRW%exB5o%4Qs

# Getting Started

```
{
  "Sid": "VisualEditor0",
  "Effect": "Allow",
  "Action": [
    "s3:GetBucketAcl",
    "s3:PutBucketAcl",
    "s3:GetObject",
    "s3:PutObject",
    "s3:DeleteObject",
    "s3:ListBucket",
    "lambda:ListFunctions",
    "lambda:GetFunction",
    "lambda:ListLayerVersions",
    "lambda:EnableReplication",
    "cloudfront:ListCloudFrontOriginAccessIdentities",
    "cloudfront:ListFunctions",
    "cloudfront:ListOriginAccessControls",
    "cloudfront:ListFieldLevelEncryptionConfigs",
    "cloudfront:ListOriginRequestPolicies",
    "cloudfront:GetDistribution",
    "cloudfront:ListDistributionsByRealtimeLogConfig",
    "cloudfront:ListKeyGroups",
    "cloudfront:ListSavingsPlans",
    "cloudfront:ListRateCards",
    "cloudfront:UpdateDistribution",
    "cloudfront:ListContinuousDeploymentPolicies",
    "cloudfront:GetDistributionConfig",
    "cloudfront:ListUsages",
    "cloudfront:ListResponseHeadersPolicies",
    "cloudfront:ListDistributionsByCachePolicyId",
    "cloudfront:ListDistributionsByLambdaFunction",
    "cloudfront:ListCachePolicies",
    "cloudfront:ListDistributionsByKeyGroup",
    "cloudfront:ListPublicKeys",
    "cloudfront:ListConflictingAliases",
    "cloudfront:ListTagsForResource",
    "cloudfront:ListRealtimeLogConfigs",
    "cloudfront:ListInvalidations",
    "cloudfront:GetInvalidation",
    "cloudfront:ListFieldLevelEncryptionProfiles",
    "cloudfront:ListDistributions",
    "cloudfront:ListStreamingDistributions",
    "cloudfront:ListKeyValueStores",
    "cloudfront:ListDistributionsByWebACLId",
    "cloudfront:ListDistributionsByResponseHeadersPolicyId",
    "cloudfront:ListDistributionsByOriginRequestPolicyId",
    "cloudfront:CreateDistribution",
    "cloudfront:TagResource",
    "cloudfront:CreateInvalidation",
    "cloudfront:UntagResource",
    "route53:ListHostedZones",
    "route53:GetHostedZone",
    "route53:ListResourceRecordSets",
    "route53:GetChange",
    "route53:ChangeResourceRecordSets",
    "route53:CreateHostedZone",
    "route53:GetDNSSEC",
    "route53:ListTagsForResource",
    "route53:UpdateHostedZoneComment",
    "route53:EnableHostedZoneDNSSEC",
    "route53:ChangeTagsForResource",
    "ec2:DescribeVpcs",
    "ec2:DescribeRegions",
    "route53:AssociateVPCWithHostedZone",
    "route53:DisassociateVPCFromHostedZone",
    "wafv2:ListWebACLs",
    "wafv2:GetWebACL",
    "wafv2:UpdateWebACL",
    "wafv2:CreateWebACL",
    "wafv2:DeleteWebACL",
    "wafv2:ListResourcesForWebACL",
    "wafv2:AssociateWebACL",
    "wafv2:DisassociateWebACL",
    "wafv2:GetSampledRequests",
    "wafv2:ListAvailableManagedRuleGroups",
    "wafv2:ListAvailableManagedRuleGroupVersions",
    "wafv2:DescribeManagedRuleGroup",
    "wafv2:ListRuleGroups",
    "wafv2:GetRuleGroup",
    "wafv2:ListIPSets",
    "wafv2:GetIPSet",
    "wafv2:ListRegexPatternSets",
    "wafv2:GetRegexPatternSet",
    "wafv2:ListTagsForResource",
    "wafv2:TagResource",
    "wafv2:UntagResource",
    "bedrock:ListFoundationModels",
    "bedrock:GetFoundationModel",
    "bedrock:InvokeModel",
    "bedrock:Converse",
    "sts:GetCallerIdentity",
    "iam:AddUserToGroup",
    "iam:AttachGroupPolicy",
    "iam:AttachRolePolicy",
    "iam:AttachUserPolicy",
    "iam:CreateGroup",
    "iam:CreatePolicy",
    "iam:CreatePolicyVersion",
    "iam:CreateRole",
    "iam:CreateUser",
    "iam:DeleteGroupPolicy",
    "iam:DeletePolicyVersion",
    "iam:DeleteRolePolicy",
    "iam:DeleteUserPolicy",
    "iam:DetachGroupPolicy",
    "iam:DetachRolePolicy",
    "iam:DetachUserPolicy",
    "iam:GetGroup",
    "iam:GetGroupPolicy",
    "iam:GetLoginProfile",
    "iam:GetPolicy",
    "iam:GetPolicyVersion",
    "iam:GetRole",
    "iam:GetRolePolicy",
    "iam:GetUser",
    "iam:GetUserPolicy",
    "iam:ListAttachedGroupPolicies",
    "iam:ListAttachedRolePolicies",
    "iam:ListAttachedUserPolicies",
    "iam:ListGroupPolicies",
    "iam:ListGroupsForUser",
    "iam:ListPolicies",
    "iam:ListPolicyTags",
    "iam:ListPolicyVersions",
    "iam:ListRolePolicies",
    "iam:ListRoleTags",
    "iam:ListUserPolicies",
    "iam:ListUserTags",
    "iam:PutGroupPolicy",
    "iam:PutRolePolicy",
    "iam:PutUserPolicy",
    "iam:RemoveUserFromGroup",
    "iam:TagPolicy",
    "iam:TagRole",
    "iam:TagUser",
    "iam:UntagPolicy",
    "iam:UntagRole",
    "iam:UntagUser",
    "iam:UpdateAssumeRolePolicy",
    "iam:UpdateRole",
    "sso:AttachCustomerManagedPolicyReferenceToPermissionSet",
    "sso:AttachManagedPolicyToPermissionSet",
    "sso:CreateAccountAssignment",
    "sso:CreatePermissionSet",
    "sso:CreateTrustedTokenIssuer",
    "sso:DeleteAccountAssignment",
    "sso:DeleteInlinePolicyFromPermissionSet",
    "sso:DescribeApplication",
    "sso:DescribeInstance",
    "sso:DescribeInstanceAccessControlAttributeConfiguration",
    "sso:DescribePermissionSet",
    "sso:DescribeTrustedTokenIssuer",
    "sso:DetachCustomerManagedPolicyReferenceFromPermissionSet",
    "sso:DetachManagedPolicyFromPermissionSet",
    "sso:GetApplicationAssignmentConfiguration",
    "sso:GetApplicationSessionConfiguration",
    "sso:GetInlinePolicyForPermissionSet",
    "sso:GetPermissionsBoundaryForPermissionSet",
    "sso:ListAccountAssignments",
    "sso:ListAccountAssignmentsForPrincipal",
    "sso:ListAccountsForProvisionedPermissionSet",
    "sso:ListApplicationAccessScopes",
    "sso:ListApplicationAuthenticationMethods",
    "sso:ListApplicationGrants",
    "sso:ListApplications",
    "sso:ListCustomerManagedPolicyReferencesInPermissionSet",
    "sso:ListInstances",
    "sso:ListManagedPoliciesInPermissionSet",
    "sso:ListPermissionSets",
    "sso:ListTagsForResource",
    "sso:ListTrustedTokenIssuers",
    "sso:ProvisionPermissionSet",
    "sso:PutApplicationAccessScope",
    "sso:PutApplicationAuthenticationMethod",
    "sso:PutApplicationGrant",
    "sso:PutApplicationSessionConfiguration",
    "sso:PutInlinePolicyToPermissionSet",
    "sso:UpdatePermissionSet",
    "sso:UpdateTrustedTokenIssuer",
    "identitystore:CreateGroup",
    "identitystore:CreateGroupMembership",
    "identitystore:CreateUser",
    "identitystore:DeleteGroupMembership",
    "identitystore:DescribeGroup",
    "identitystore:DescribeUser",
  ],
  "Resource": "*"
}
```