AWS MARKETPLACE · CUSTOMER CHECKLIST

i2i Intelligence Platform Customer Pre-Deployment Checklist

Complete every item **before** launching the CloudFormation template. These one-time steps prepare the AWS resources the stack requires.

⏱ **Estimated time:** 30–60 minutes (one-time setup) ECS Fargate CloudFormation

🌐 **Supported regions:** us-east-1 · us-east-2 · us-west-2 · eu-west-1 · eu-central-1 · ap-southeast-1 · ap-northeast-1

🤖 **AI model:** Claude Sonnet 4.5 via Amazon Bedrock

CONTENTS

PREREQUISITES

1 AWS Account & Region
2 Networking
3 Security Groups
4 TLS Certificate
5 S3 Bucket
6 Secrets Manager
7 Cube Cloud Details
8 Amazon MWAA
9 Bedrock Model Access

1 AWS Account & Region REQUIRED

Choose your deployment region and verify IAM permissions

Choose a region

All resources — VPC, subnets, S3 bucket, ACM certificate, Secrets Manager secrets, ECS cluster, and CloudWatch logs — must reside in the **same AWS region**. Mixing regions will cause the deployment to fail.

Geography	Supported Region IDs
United States	`us-east-1` `us-east-2` `us-west-2`
Europe	`eu-west-1` `eu-central-1`
Asia Pacific	`ap-southeast-1` `ap-northeast-1`

> ⚠️ **us-west-1 is not supported.** Select the nearest available region from the list above. Throughout this guide, your chosen region is referred to as `<YOUR-REGION>`.

Required IAM permissions

The IAM user or role used to launch the stack requires the following permissions. The simplest option is to attach the **AdministratorAccess** managed policy during initial deployment.

Service	Actions required
CloudFormation	`cloudformation:*`
IAM	`iam:CreateRole` · `iam:PutRolePolicy` · `iam:AttachRolePolicy` · `iam:PassRole`
EC2	`ec2:CreateSecurityGroup` · `ec2:AuthorizeSecurityGroupIngress`
ECS	`ecs:*`
Elastic Load Balancing	`elasticloadbalancing:*`
CloudWatch Logs	`logs:CreateLogGroup`
Secrets Manager	`secretsmanager:GetSecretValue`

---

2 Networking REQUIRED

VPC and subnets for ALB, ECS tasks, and MWAA

You need a single VPC containing both public subnets (for the ALB and ECS Fargate tasks) and private subnets (for Amazon MWAA). ECS tasks, although launched in public subnets, are protected by a security group that only permits traffic from the ALB.

VPC requirements

- [ ] VPC in `<YOUR-REGION>` with **DNS resolution** enabled (VPC → Actions → Edit DNS resolution → enable)
- [ ] VPC with **DNS hostnames** enabled (VPC → Actions → Edit DNS hostnames → enable)
- [ ] Note the **VPC ID** (format: `vpc-xxxxxxxxxxxxxxxxx`)
- [ ] At least **two public subnets** in different Availability Zones — used by the ALB and ECS tasks. Each must have an Internet Gateway route in its route table.
- [ ] At least **two private subnets** in different Availability Zones — used by MWAA workers. These require a NAT Gateway route for outbound internet access.

- [ ] Note all **subnet IDs** (format: `subnet-xxxx` , `subnet-yyyy` , ...)

> 💡 **Creating from scratch?** Go to AWS Console → VPC → *Create VPC* → choose *VPC and more*. Select 2 AZs, 2 public subnets, 2 private subnets, and 1 NAT Gateway (1 AZ is sufficient for evaluation).

---

3 Security Groups REQUIRED

Must be pre-created before launching the CloudFormation stack

Create the two security groups below and supply their IDs as CloudFormation parameters. This gives you explicit control over which IP addresses and services can reach the platform.

Security Group 1 — ALB `i2i-alb-sg`

Direction	Protocol	Port	Source / Destination	Purpose
Inbound	TCP	80	*Your users' IP CIDR*	HTTP access
Inbound	TCP	443	*Your users' IP CIDR*	HTTPS access
Outbound	All	All	0.0.0.0/0	Required to reach ECS tasks

Replace *your users' IP CIDR* with your office/VPN CIDR, e.g. `203.0.113.0/24` . Use `0.0.0.0/0` only for public internet access.

Security Group 2 — ECS Tasks `i2i-ecs-sg`

Direction	Protocol	Port	Source / Destination	Purpose
Inbound	TCP	80	ALB security group ID	ALB to UI container
Inbound	All	All	This security group (self)	Inter-service traffic
Outbound	All	All	0.0.0.0/0	Calls to Cube Cloud, Bedrock, S3

> ⚠️ For the ALB inbound rule on the ECS SG, specify the **security group ID** (`sg-xxx`) of the ALB SG — not a CIDR block.

CLI — create both security groups

```bash
# ── Create ALB security group ──────────────────────────────
ALB_SG=$(aws ec2 create-security-group \
  --group-name i2i-alb-sg \
  --description "i2i ALB - inbound HTTP and HTTPS from users" \
  --vpc-id <your-vpc-id> \
  --region <YOUR-REGION> \
  --query GroupId --output text)

aws ec2 authorize-security-group-ingress \
  --group-id $ALB_SG --protocol tcp --port 80 \
  --cidr <your-cidr> --region <YOUR-REGION>

aws ec2 authorize-security-group-ingress \
  --group-id $ALB_SG --protocol tcp --port 443 \
  --cidr <your-cidr> --region <YOUR-REGION>

echo "ALB SG ID: $ALB_SG"    # ← note this value

# ── Create ECS tasks security group ────────────────────────
ECS_SG=$(aws ec2 create-security-group \
  --group-name i2i-ecs-sg \
  --description "i2i ECS tasks - inbound from ALB and inter-service" \
  --vpc-id <your-vpc-id> \
  --region <YOUR-REGION> \
  --query GroupId --output text)

aws ec2 authorize-security-group-ingress \
  --group-id $ECS_SG --protocol tcp --port 80 \
  --source-group $ALB_SG --region <YOUR-REGION>

aws ec2 authorize-security-group-ingress \
  --group-id $ECS_SG --protocol -1 \
  --source-group $ECS_SG --region <YOUR-REGION>

echo "ECS SG ID: $ECS_SG"    # ← note this value
```

- [ ] Note ALB Security Group ID → `ALBSecurityGroupId` CloudFormation parameter
- [ ] Note ECS Security Group ID → `ECSSecurityGroupId` CloudFormation parameter

4

TLS Certificate OPTIONAL

Strongly recommended for production; leave blank for HTTP-only evaluation

When provided, the ALB automatically redirects all HTTP traffic to HTTPS. If this parameter is left blank, the platform runs on HTTP only.

1 **Request a public certificate in ACM**

AWS Console → Certificate Manager → *Request* → *Request public certificate*.
Enter your domain (e.g. `i2i.yourcompany.com` or `*.yourcompany.com`). Choose **DNS validation** and add the supplied CNAME record to your DNS. Wait for status to show **Issued**.

2 **Note the certificate ARN**

Format: `arn:aws:acm:<YOUR-REGION>:<account-id>:certificate/<uuid>`

> **Important:** ACM certificates are region-specific. The certificate *must* be requested in the same region as your deployment. Certificates from other regions cannot be attached to the ALB.

---

5 S3 Bucket REQUIRED

Stores all pipeline run outputs — charts, reports, metadata

i2i writes all pipeline results to a dedicated S3 bucket in your account. The bucket starts empty; the platform automatically creates the required folder structure and JSON schema files on first deployment.

```bash
# For all regions except us-east-1
aws s3api create-bucket \
  --bucket <your-unique-bucket-name> \
  --region <YOUR-REGION> \
  --create-bucket-configuration LocationConstraint=<YOUR-REGION>

# For us-east-1 only (omit --create-bucket-configuration)
aws s3api create-bucket \
  --bucket <your-unique-bucket-name> \
  --region us-east-1
```

- [ ] Bucket created in `<YOUR-REGION>`
- [ ] **Block Public Access** — all four settings enabled (this is the default; do not disable)
- [ ] Note the bucket name → `S3BucketName` CloudFormation parameter

> Do **not** upload anything to this bucket before launching the stack. The one-time setup task generates `metadata.json` and `new_test.json` automatically during deployment.

---

6 AWS Secrets Manager REQUIRED

Two secrets must be created — credentials are never stored in CloudFormation parameters

Secret 1 — Cube.js API Secret Token

```bash
aws secretsmanager create-secret \
  --region <YOUR-REGION> \
  --name "i2i/cubejs-api-secret" \
  --secret-string "<your-cubejs-api-secret-token>"
```

Secret 2 — Cube DB Password

```bash
aws secretsmanager create-secret \
  --region <YOUR-REGION> \
  --name "i2i/cube-db-password" \
  --secret-string "<your-cube-db-password>"
```

⚠️ Copy the full ARN from each command's response — **including the random suffix**. Example format: `arn:aws:secretsmanager:us-west-2:123456789012:secret:i2i/cubejs-api-secret-AbCdEf`. The ARN with suffix is required as the CloudFormation parameter value.

- [ ] Note secret ARN for `i2i/cubejs-api-secret` → `CubeJsApiSecretArn` parameter
- [ ] Note secret ARN for `i2i/cube-db-password` → `CubeDbPasswordSecretArn` parameter

---

7 Cube Cloud Connection Details REQUIRED

Collect from the Cube Cloud console before launching the stack

i2i connects to your Cube Cloud deployment as its semantic layer via both the REST API and the SQL API. All four values below are required CloudFormation parameters.

CloudFormation Parameter	Where to find it	Example value
`CubeApiUrl`	Cube Cloud → Deployment → Overview → *API URL*	`https://<deployment>.gcp-us-central1.cubecloudapp.dev/cubejs-api/v1`
`CubeDbName`	Subdomain part of your Cube deployment URL	`grateful-tahr`
`CubeDbUser`	Cube Cloud → Deployment → SQL API credentials	`cube` (default)
`CubeDbHost`	Cube Cloud → Deployment → Overview → *SQL API endpoint*	`<deployment>.sql.gcp-us-central1.cubecloudapp.dev`

ℹ The Cube.js API secret token and Cube DB password are supplied via Secrets Manager ARNs from Step 6 — not entered directly as parameters.

---

8 Amazon MWAA REQUIRED

Decide the environment name now — create the environment after CloudFormation completes

🚫 **Do not create the MWAA environment yet.** Only decide and record the name you intend to use (e.g. `i2i-prod`). The environment must be created *after* the CloudFormation stack reaches `CREATE_COMPLETE` — you will need output values from the stack when configuring it.

What to prepare now

- [ ] Decide the MWAA environment name (e.g. `i2i-prod`) → `MwaaEnvName` CloudFormation parameter
- [ ] Ensure the private subnets from Step 2 are in the same VPC — MWAA will use them
- [ ] Have a separate S3 bucket ready for MWAA DAG files (distinct from the i2i working bucket in Step 5)
- [ ] Leave the `MwaaRegion` CloudFormation parameter **blank** if MWAA and the ECS stack are in the same region

S3 bucket separation — important

Bucket	Purpose	Contents
MWAA DAGs bucket	Airflow file storage	`startup.sh` , `dags/airflow_backend.py`
i2i working bucket	Pipeline run outputs	Charts, EDA results, reports, `metadata.json`

MWAA execution role permissions

After creating the MWAA environment, add the following inline policy to its execution role (IAM → Roles → the role attached to your MWAA environment → Add inline policy):

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "MwaaMetrics",
      "Effect": "Allow",
      "Action": "airflow:PublishMetrics",
      "Resource": "arn:aws:airflow:<YOUR-REGION>:<account-id>:environment/<your-mwaa-env-name>"
    },
    {
      "Sid": "S3Access",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*", "s3:GetBucket*", "s3:List*",
        "s3:PutObject", "s3:PutObjectAcl", "s3:DeleteObject",
        "s3:GetEncryptionConfiguration"
      ],
      "Resource": [
        "arn:aws:s3:::<your-mwaa-dags-bucket>",
        "arn:aws:s3:::<your-mwaa-dags-bucket>/*",
        "arn:aws:s3:::<your-i2i-working-bucket-name>",
        "arn:aws:s3:::<your-i2i-working-bucket-name>/*"
      ]
    },
    {
      "Sid": "CloudWatchLogs",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream", "logs:CreateLogGroup", "logs:PutLogEvents",
        "logs:GetLogEvents", "logs:GetLogRecord", "logs:GetLogGroupFields",
        "logs:GetQueryResults"
      ],
      "Resource": "arn:aws:logs:<YOUR-REGION>:<account-id>:log-group:airflow-<your-mwaa-env-name>-*"
    },
    {
      "Sid": "CloudWatchDescribeAndMetrics",
      "Effect": "Allow",
      "Action": ["logs:DescribeLogGroups", "cloudwatch:PutMetricData"],
      "Resource": "*"
    },
    {
      "Sid": "SQSCelery",
      "Effect": "Allow",
      "Action": [
        "sqs:ChangeMessageVisibility", "sqs:DeleteMessage",
        "sqs:GetQueueAttributes", "sqs:GetQueueUrl",
        "sqs:ReceiveMessage", "sqs:SendMessage"
      ],
      "Resource": "arn:aws:sqs:*:*:airflow-celery-*"
```

```
    },
    {
      "Sid": "KMS",
      "Effect": "Allow",
      "Action": ["kms:Decrypt", "kms:DescribeKey", "kms:GenerateDataKey*", "kms:Encrypt"],
      "Resource": "*"
    }
  ]
}
```

⚠️ Both S3 buckets must be listed in the policy — the MWAA DAGs bucket (for DAG file access) and the i2i working bucket (for pipeline outputs). Omitting either will cause pipeline failures.

ℹ️ When you create the MWAA environment later: place it in the **same VPC and region** as the ECS stack, using the **private subnets** from Step 2. Enable **all log types at INFO level** in the Monitoring section.

---

9 Amazon Bedrock Model Access REQUIRED

Access must be granted before the platform can process any request

Request access to the following model in `<YOUR-REGION>` :

MODEL ID

```
us.anthropic.claude-sonnet-4-5-20250929-v1:0
```

1 Open the Bedrock console

AWS Console → **Amazon Bedrock** → *Model access* (left sidebar)

2 Request access

Click *Manage model access* → locate **Anthropic / Claude Sonnet 4.5** → check the box → click *Request model access* → *Submit*.

3 Wait for approval

Refresh until the model status shows **Access granted**. Anthropic models are approved within minutes in most regions.

⚠️ Bedrock model availability varies by region. If Claude Sonnet 4.5 is not listed in your chosen region, select the nearest supported region or contact i2i support for an alternative model ID.

i2i Intelligence Platform — Customer Pre-Deployment Checklist

Deployment and post-deployment steps are covered in the *i2i Customer Installation Guide*.