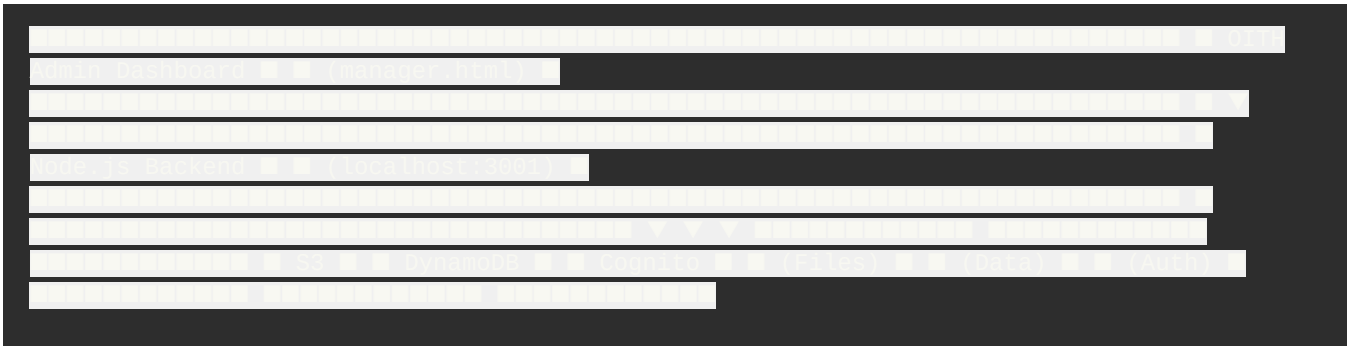
AWS Setup Guide for OITH Admin Backend

This guide walks you through setting up AWS services to store your admin data in the cloud.

Overview



Step 1: Create AWS Account

1. Go to https://aws.amazon.com
2. Click "Create an AWS Account"
3. Follow the signup process (requires credit card)
4. Enable MFA for security

Step 2: Create IAM User

1. Go to AWS Console → IAM → Users
2. Click "Add users"
3. Username: `oith-admin-backend`
4. Select "Access key - Programmatic access"
5. Attach policies:
- `AmazonS3FullAccess`
- `AmazonDynamoDBFullAccess`
6. Save the Access Key ID and Secret Access Key

Step 3: Create S3 Bucket

1. Go to AWS Console → S3
2. Click "Create bucket"
3. Bucket name: `oith-admin-documents` (must be globally unique)
4. Region: `us-east-1` (or your preferred region)
5. Block Public Access: Keep all blocked
6. Click "Create bucket"

Configure CORS (for browser uploads):

```
[ { "AllowedHeaders": ["*"], "AllowedMethods": ["GET", "PUT", "POST", "DELETE"],
"AllowedOrigins": ["http://localhost:5500", "http://localhost:3001"], "ExposeHeaders":
["ETag"] } ]
```

Step 4: Create DynamoDB Table

1. Go to AWS Console → DynamoDB
2. Click "Create table"
3. Table name: `oith-admin-data`
4. Partition key: `pk` (String)
5. Sort key: `sk` (String)
6. Settings: On-demand capacity (pay per request)
7. Click "Create table"

Step 5: Configure Backend

Navigate to the server folder:
`bash cd "C:\Users\mattr\OneDrive\Desktop\Ross, Matt\Operations\MBA\OITH\prototype\server"`

Copy the environment template:
`bash copy env-template.txt .env`

Edit `.env` with your AWS credentials:
`AWS_ACCESS_KEY_ID=AKIA...your-key AWS_SECRET_ACCESS_KEY=your-secret-key AWS_REGION=us-east-1 AWS_S3_BUCKET=oith-admin-documents AWS_DYNAMODB_TABLE=oith-admin-data PORT=3001`

Install dependencies:
`bash npm install`

Start the server:
`bash npm start`

Step 6: Migrate Existing Data

Once the server is running, you can migrate your localStorage data to AWS:

1. Open `http://localhost:3001/api/health` to verify server is running
2. In the admin dashboard, there will be a "Migrate to AWS" button
3. Click it to transfer all local data to cloud storage

Or use the API directly:

```
// In browser console on manager.html const localData = {}; for (let i = 0; i < localStorage.length; i++) { const key = localStorage.key(i); if (key.startsWith('oith_')) { localData[key] = localStorage.getItem(key); } } fetch('http://localhost:3001/api/sync/migrate-from-local', { method: 'POST', headers: { 'Content-Type': 'application/json' }, body: JSON.stringify(localData) }).then(r => r.json()).then(console.log);
```

API Endpoints

Endpoint	Method	Description
`/api/health`	GET	Health check
`/api/users`	GET/POST	User management
`/api/documents/:category/:itemId`	GET/POST/DELETE	Document uploads
`/api/experiments/active`	GET/POST/PUT	Active experiments

`/api/experiments/history`	GET/POST	Experiment history
`/api/org`	GET/PUT	Organization data
`/api/payroll`	GET/PUT	Payroll data
`/api/sync/export`	GET	Export all data
`/api/sync/import`	POST	Import data

Cost Estimate

Service	Free Tier	After Free Tier
S3	5GB storage, 20K requests	~$0.023/GB
DynamoDB	25GB, 25 read/write units	Pay per request
Lambda (if used)	1M requests/month	$0.20 per 1M

Estimated monthly cost for small usage: $1-5

Troubleshooting

"Access Denied" errors

- Check IAM user has correct policies attached
- Verify Access Key and Secret are correct in `.env`

"Bucket not found"

- Ensure bucket name in `.env` matches exactly
- Bucket names are globally unique - yours might need a suffix

CORS errors

- Add your domain to S3 bucket CORS configuration

- Ensure backend CORS includes your frontend URL

Connection timeouts

- Check AWS region matches in all configurations
- Verify network/firewall allows AWS connections

Security Best Practices

1. **Never commit `.env` to git** - Add to `.gitignore`
2. **Use environment variables** in production
3. **Rotate access keys** periodically
4. **Enable CloudTrail** for audit logging
5. **Set up billing alerts** to avoid surprises

Next Steps

After basic setup:
1. Set up AWS CloudWatch for monitoring
2. Configure S3 lifecycle rules for document archival
3. Enable DynamoDB point-in-time recovery
4. Consider AWS Cognito for admin authentication
5. Set up CI/CD for automated deployments