# Javumbo Serverless Backend

This directory contains the serverless refactoring of the Javumbo flashcard application.

## Structure

```
server_lambda/
├── terraform/          # Infrastructure as Code (AWS resources)
├── src/               # Lambda function source code
├── tests/             # Unit tests for core functionality
├── docs/              # Documentation
└── requirements.txt   # Python dependencies
```

## Quick Start

### Day 1: Deploy Infrastructure

```
cd terraform
terraform init
terraform plan
terraform apply

# Save outputs
terraform output > ../infrastructure.txt
```

### Day 2: Test S3SQLiteConnection

```
# Set environment variable
cd terraform
export S3_BUCKET=$(terraform output -raw s3_bucket_name)

# Run tests
cd ../tests
./run_tests.sh
```

## Day 2: S3SQLiteConnection Testing

**Objective**: Prove S3 SQLite pattern works (download → modify → upload).

### Manual Test Run

```
# Set bucket name
export S3_BUCKET=$(cd terraform && terraform output -raw s3_bucket_name)
```

```
# Test 2.1: New user database creation
cd tests
python3 test_s3_sqlite_new_user.py

# Test 2.2: Read/write persistence
python3 test_s3_sqlite_readwrite.py

# Or run all tests at once:
./run_tests.sh
```

### Expected Results

**Test 2.1**: Creates new Anki database with proper schema

- ✅ Tables created: col, cards, notes, revlog
- ✅ Database uploaded to S3
- ✅ Can be re-opened

**Test 2.2**: Data persists across connections

- ✅ Data written in first connection
- ✅ Data uploaded to S3
- ✅ Data readable in second connection

## Development

### Local Testing (without Lambda)

The tests run directly on your machine using your AWS credentials, simulating what Lambda will do.

### Requirements

- Python 3.11+
- AWS CLI configured
- boto3 installed (`pip install boto3`)

### Environment Variables

```
# Required
export S3_BUCKET=javumbo-user-dbs-{account-id}

# Optional (set by Lambda automatically)
export DYNAMODB_USERS_TABLE=javumbo-users
export DYNAMODB_LOCKS_TABLE=javumbo-user-locks
export SECRET_KEY=your-secret-key
```

## Progress

- ✅ Day 1: Infrastructure deployed (S3, DynamoDB, Lambda, API Gateway)
- ✅ Day 2: S3SQLiteConnection implementation (baseline: 513ms avg latency)
- ✅ Day 3: Lambda container caching (35% latency reduction, 98% cache hit rate, 2.07-2.41x speedup)
- ✅ Day 4: Optimistic locking with ETags (ZERO data loss, ConflictError on concurrent writes)
- ✅ Day 5: DynamoDB user repository (UserRepository with bcrypt, 100% test pass rate, /tmp cleanup utility)

## Documentation

- Week 1 Plan - Detailed Day 1-5 plan
- Main Refactoring Plan - Full 20-day plan
- Terraform README - Infrastructure details

## Troubleshooting

### Test fails with "NoSuchBucket"

```
# Verify bucket exists
aws s3 ls | grep javumbo

# Re-export bucket name
export S3_BUCKET=$(cd terraform && terraform output -raw s3_bucket_name)
```

### Test fails with "AccessDenied"

```
# Verify AWS credentials
aws sts get-caller-identity

# Check bucket permissions
aws s3api get-bucket-policy --bucket $S3_BUCKET
```

### Import error: "No module named s3_sqlite"

```
# Tests add ../src to path automatically
# If running interactively:
export
PYTHONPATH=/Users/emadruga/proj/javumbo/server_lambda/src:$PYTHONPATH
```

## Next Steps

After Week 1 completion:

- **Week 2**: Flask API migration to Lambda handlers