KINTO QA Management System

macOS Deployment Guide

Complete guide for deploying on your MacBook

Last Updated: November 4, 2025

Ø=ÜË Table of Contents

1. [Prerequisites](#prerequisites)
2. [Step 1: Install Required Software](#step-1-install-required-software)
3. [Step 2: Download the Application](#step-2-download-the-application)
4. [Step 3: Set Up PostgreSQL Database](#step-3-set-up-postgresql-database)
5. [Step 4: Configure Environment Variables](#step-4-configure-environment-variables)
6. [Step 5: Install Dependencies](#step-5-install-dependencies)
7. [Step 6: Initialize Database](#step-6-initialize-database)
8. [Step 7: Start the Application](#step-7-start-the-application)
9. [Step 8: Access Your Application](#step-8-access-your-application)
10. [Troubleshooting](#troubleshooting)
11. [Stopping the Application](#stopping-the-application)

Prerequisites

Before you begin, ensure you have:

- ' macOS 10.15 (Catalina) or later
- ' Administrator access to your MacBook
- ' Internet connection
- ' Basic familiarity with Terminal

Step 1: Install Required Software

1.1 Install Homebrew (Package Manager)

Open **Terminal** (found in Applications > Utilities) and run:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/
install.sh)"
```

Follow the on-screen instructions. After installation, verify:

```
brew --version
```

1.2 Install Node.js 20+

```
brew install node@20
```

Verify installation:

```
node --version  # Should show v20.x.x or higher
npm --version   # Should show 10.x.x or higher
```

1.3 Install PostgreSQL 14+

```
brew install postgresql@14
```

Start PostgreSQL service:

```
brew services start postgresql@14
```

Verify PostgreSQL is running:

```
psql --version  # Should show PostgreSQL 14.x or higher
```

1.4 Install Git (Optional, for downloading code)

```
brew install git
```

Step 2: Download the Application

Option A: Download from Replit (Recommended)

1. Download as ZIP:
  • In your Replit workspace, click the three dots menu ("î)
  • Select "Download as ZIP"
  • Save the file to your Downloads folder

2. Extract the ZIP file:
   `bash
   cd ~/Downloads
   unzip kinto-qa-system.zip
   cd kinto-qa-system
   `

Option B: Clone from Git (If you've pushed to GitHub)

```
cd ~/Documents
git clone https://github.com/yourusername/kinto-qa-system.git
cd kinto-qa-system
```

Option C: Manual File Transfer

1. Create a project folder:
   `bash
   mkdir -p ~/Documents/kinto-qa-system
   cd ~/Documents/kinto-qa-system
   `

2. Copy all files from your Replit workspace to this folder

Step 3: Set Up PostgreSQL Database

3.1 Create Database User

Open Terminal and access PostgreSQL:

```
psql postgres
```

Create a database user (you'll be prompted to create a password):

```
CREATE USER kinto_admin WITH PASSWORD 'your_secure_password';
ALTER USER kinto_admin CREATEDB;
\q
```

3.2 Create Database

```
createdb -U kinto_admin kinto_qa_db
```

If you get a permission error, run as postgres user:

```
psql postgres -c "CREATE DATABASE kinto_qa_db OWNER kinto_admin;"
```

3.3 Verify Database Connection

```
psql -U kinto_admin -d kinto_qa_db -c "SELECT version();"
```

You should see PostgreSQL version information.

Step 4: Configure Environment Variables

4.1 Create `.env` File

In your project directory, create a .env file:

```
cd ~/Documents/kinto-qa-system
touch .env
```

4.2 Edit `.env` File

Open the file with TextEdit or your preferred editor:

```
open -e .env
```

Add the following content (replace values as needed):

```
# Database Configuration
DATABASE_URL=postgresql://kinto_admin:your_secure_password@localhost:5432/kinto_qa_db
PGHOST=localhost
PGPORT=5432
PGUSER=kinto_admin
PGPASSWORD=your_secure_password
PGDATABASE=kinto_qa_db
# Session Secret (generate a random string)
SESSION_SECRET=your-super-secret-random-string-change-this
# Node Environment
NODE_ENV=production
PORT=5000
```

Important:
• Replace `your_secure_password` with the password you created in Step 3.1
• Replace `your-super-secret-random-string-change-this` with a random string (at least 32 characters)

4.3 Generate Session Secret (Optional but Recommended)

Generate a secure random string:

```
node -e "console.log(require('crypto').randomBytes(32).toString('hex'))"
```

Copy the output and use it as your SESSION_SECRET.

Step 5: Install Dependencies

Navigate to your project directory and install all required packages:

```
cd ~/Documents/kinto-qa-system
npm install
```

This will install all dependencies listed in package.json. It may take a few minutes.

Step 6: Initialize Database

6.1 Import Database Schema

Download the database_schema.sql file from your Replit deployment (available at /download.html), then import it:

```
psql -U kinto_admin -d kinto_qa_db -f database_schema.sql
```

This will create all tables, indexes, and seed data including:
- Default roles (admin, manager, operator, reviewer)
- Default admin user (username: admin, password: Admin@123)
- Sample units of measure
- Sample machine types
- Role permissions

6.2 Verify Database Setup

Check if tables were created:

```
psql -U kinto_admin -d kinto_qa_db -c "\dt"
```

You should see a list of tables including:
- users
- roles
- role_permissions
- machines
- checklist_templates
- spare_parts_catalog
- etc.

6.3 Verify Default Admin User

```
psql -U kinto_admin -d kinto_qa_db -c "SELECT username, email, first_name, last_name FROM
users WHERE username = 'admin';"
```

You should see the default admin user.

Step 7: Start the Application

7.1 Build the Application

```
npm run build
```

This compiles the frontend and backend code.

7.2 Start Production Server

```
npm start
```

You should see:

```
[express] serving on port 5000
```

7.3 Alternative: Development Mode

For development with hot-reload:

```
npm run dev
```

Step 8: Access Your Application

8.1 Open in Browser

Open your web browser and navigate to:

```
http://localhost:5000
```

8.2 Login

Use the default admin credentials:

• Username: `admin`
• Password: `Admin@123`

& þ IMPORTANT: Change this password immediately after your first login!

8.3 First-Time Setup

After logging in:

1. Go to User Management and change the admin password
2. Create additional users as needed
3. Configure roles and permissions in Role Management
4. Set up your machines in Machines
5. Create checklist templates
6. Start using the system!

Troubleshooting

Issue: Port 5000 Already in Use

If you see Error: listen EADDRINUSE :::5000:

```
# Find process using port 5000
lsof -ti:5000
# Kill the process (replace PID with the number from above)
kill -9 PID
```

Or change the port in .env:

```
PORT=3000
```

Issue: Cannot Connect to Database

Error: `connection refused` or `password authentication failed`

Solution:

1. Verify PostgreSQL is running:
   `bash
   brew services list | grep postgresql
   `

2. Restart PostgreSQL:
   `bash
   brew services restart postgresql@14
   `

3. Check your DATABASE_URL in `.env` file
4. Verify database exists:
   `bash
   psql -U kinto_admin -l
   `

Issue: Module Not Found

Error: `Cannot find module 'xyz'`

```
rm -rf node_modules package-lock.json
npm install
```

Issue: Database Schema Import Fails

Error: During `psql -f database_schema.sql`

Solution:
1. Drop and recreate the database:
   `bash
   dropdb -U kinto_admin kinto_qa_db
   createdb -U kinto_admin kinto_qa_db
   `

2. Re-import the schema:
   `bash
   psql -U kinto_admin -d kinto_qa_db -f database_schema.sql
   `

Issue: Permission Denied (EACCES)

Solution:
Fix npm permissions:

```
sudo chown -R $USER /usr/local/lib/node_modules
```

Getting Help

If you encounter issues not covered here:

1. Check the application logs in Terminal
2. Review the `DEPLOYMENT_GUIDE.md` for general deployment issues
3. Check PostgreSQL logs: `tail -f /usr/local/var/log/postgresql@14.log`

Stopping the Application

Stop Development Server

Press Ctrl + C in the Terminal where the app is running.

Stop PostgreSQL Service

```
brew services stop postgresql@14
```

Restart PostgreSQL Service

```
brew services restart postgresql@14
```

Running on System Startup (Optional)

Auto-start PostgreSQL

PostgreSQL is already configured to auto-start via Homebrew services.

Auto-start Application

Create a launch agent:

```
mkdir -p ~/Library/LaunchAgents
```

Create file ~/Library/LaunchAgents/com.kinto.qa.plist:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Label</key>
    <string>com.kinto.qa</string>
    <key>ProgramArguments</key>
    <array>
        <string>/usr/local/bin/node</string>
        <string>/Users/YOUR_USERNAME/Documents/kinto-qa-system/dist/index.js</string>
    </array>
    <key>WorkingDirectory</key>
    <string>/Users/YOUR_USERNAME/Documents/kinto-qa-system</string>
    <key>RunAtLoad</key>
    <true/>
    <key>KeepAlive</key>
    <true/>
    <key>StandardOutPath</key>
    <string>/tmp/kinto-qa.log</string>
    <key>StandardErrorPath</key>
    <string>/tmp/kinto-qa-error.log</string>
</dict>
</plist>
```

Replace YOUR_USERNAME with your macOS username.

Load the agent:

```
launchctl load ~/Library/LaunchAgents/com.kinto.qa.plist
```

Accessing from Other Devices on Your Network

Find Your Mac's IP Address

```
ipconfig getifaddr en0  # For WiFi
ipconfig getifaddr en1  # For Ethernet
```

Example output: 192.168.1.100

Access from Mobile/Tablet

On your mobile device or tablet connected to the same WiFi network, open:

```
http://192.168.1.100:5000
```

Replace 192.168.1.100 with your Mac's IP address.

Enable macOS Firewall (Recommended)

1. Go to System Preferences > Security & Privacy > Firewall
2. Click "Firewall Options"
3. Add Node.js to allowed applications

Updating the Application

Pull Latest Changes (If using Git)

```
cd ~/Documents/kinto-qa-system
git pull origin main
npm install
npm run build
npm start
```

Manual Update

1. Download the latest version from Replit
2. Extract to a new folder
3. Copy your `.env` file to the new folder
4. Run `npm install` and `npm run build`
5. Start the application

Backup and Maintenance

Backup Database

```
# Create backup directory
mkdir -p ~/kinto-backups
# Backup database
pg_dump -U kinto_admin kinto_qa_db > ~/kinto-backups/backup-$(date +%Y%m%d-%H%M%S).sql
```

Restore Database

```
psql -U kinto_admin -d kinto_qa_db -f ~/kinto-backups/backup-YYYYMMDD-HHMMSS.sql
```

Automated Daily Backups

Add to crontab:

```
crontab -e
```

Add this line (runs daily at 2 AM):

```
0 2 * * * pg_dump -U kinto_admin kinto_qa_db > ~/kinto-backups/backup-$(date +\%Y\%m\%d-\
%H\%M\%S).sql
```

System Requirements

Minimum:
• macOS 10.15+
• 4 GB RAM
• 2 GB free disk space
• Node.js 20+
• PostgreSQL 14+

Recommended:
• macOS 11.0 (Big Sur or later)
• 8 GB RAM
• 10 GB free disk space
• SSD storage
• Stable internet connection (for initial setup)

Security Best Practices

1. ' Change default admin password immediately
2. ' Use strong passwords for database users
3. ' Keep your macOS and software updated
4. ' Enable macOS Firewall
5. ' Regular database backups
6. ' Don't expose port 5000 to the internet without proper security
7. ' Use environment variables for sensitive data (never commit `.env` to git)

Additional Resources

• Main Deployment Guide: `DEPLOYMENT_GUIDE.md`
• System Design: `SYSTEM_DESIGN.md`
• Database Schema: `database_schema.sql`
• Mobile Guides: `MOBILE_APK_BUILD_GUIDE.md`, `MOBILE_IPA_BUILD_GUIDE.md`

Support

For issues or questions:

1. Review this guide and troubleshooting section
2. Check application logs in Terminal
3. Review PostgreSQL logs
4. Consult the complete `DEPLOYMENT_GUIDE.md`

Ø<ß‰ Congratulations! Your KINTO QA Management System is now running on your MacBook!

Access it at: **http://localhost:5000**

Default credentials:
• Username: `admin`
• Password: `Admin@123`

Remember to change the default password immediately!