📦 Deployment Package Summary

✅ What's Been Prepared

Your trading application has been located and packaged for deployment!

📂 Application Location

Source Directory: `/home/ubuntu/code_artifacts/trading-app/`

Application Structure:

```
trading-app/
├── backend/                 # Go backend server
│   ├── cmd/                 # Main application entry point
│   ├── internal/            # Internal packages (handlers, models, etc.)
│   ├── pkg/                 # Public packages
│   └── go.mod               # Go dependencies
├── frontend/                # Next.js frontend
│   ├── app/                 # Next.js app directory (pages & routing)
│   ├── components/          # React components
│   ├── lib/                 # Utility libraries
│   ├── public/              # Static assets
│   └── package.json         # Node.js dependencies
├── deploy.sh                # Automated deployment script
├── start.sh                 # Start services
├── stop.sh                  # Stop services
├── restart.sh               # Restart services
├── status.sh                # Check service status
├── logs.sh                  # View logs
└── README.md                # Application documentation
```

📦 Deployment Package

File: `trading-app-deployment.tar.gz`

Location: `/home/ubuntu/code_artifacts/trading-app-deployment.tar.gz`

Size: 349 KB (compressed)

Contains: Complete application with all source code and configuration scripts

---

📋 Files Created for Deployment

1. Deployment Guide

File: `/home/ubuntu/code_artifacts/deployment_guide.md`

Description: Comprehensive step-by-step guide with:
- Exact copy-paste commands for every step
- Explanations for each command
- Verification steps after each action
- Troubleshooting section
- Security best practices
- Quick reference commands

Format: Available in Markdown (.md) and PDF formats

---

2. Deployment Package

File: `/home/ubuntu/code_artifacts/trading-app-deployment.tar.gz`

Description: Compressed archive containing the entire application

How to Transfer:

```
scp trading-app-deployment.tar.gz root@67.211.219.94:/root/
```

🔧 Backend Details

Language: Go 1.21+

Main File: `backend/cmd/main.go`

Key Dependencies:
- `gorilla/mux` - HTTP routing
- `gorilla/websocket` - WebSocket support
- `mattn/go-sqlite3` - SQLite database
- `golang-jwt/jwt/v5` - JWT authentication
- `otiai10/gosseract` - OCR for document processing
- `xuri/excelize` - Excel file handling

Port: 8080

Features:
- RESTful API
- WebSocket for real-time updates
- Authentication with JWT
- File processing (CSV, Excel, PDF)
- Trading strategy backtesting
- Portfolio management
- AI chat integration (Abacus.AI)
- OpenAlgo integration for trading

🎨 Frontend Details

Framework: Next.js 14

Language: TypeScript

Key Dependencies:
- `next` 14.0.4 - Next.js framework
- `react` 18.2.0 - React library
- `axios` - HTTP client
- `recharts` - Data visualization
- `react-dropzone` - File uploads
- `zustand` - State management
- `tailwindcss` - CSS framework
- `lucide-react` - Icon library

Port: 3000

Features:
- Responsive design
- Real-time updates via WebSocket
- Interactive charts
- File upload/download
- Trading dashboard
- Strategy builder
- AI chat assistant
- Portfolio tracking

🖥️ VPS Requirements

Target VPS:
- IP Address: `67.211.219.94`
- OS: Ubuntu (20.04 LTS or later)
- RAM: 2GB (sufficient)
- CPU: 1 core (sufficient)
- Disk: ~5GB needed

Pre-installed:
- ✅ Go (already on VPS)

To Be Installed:
- Node.js 18.x
- npm 9.x
- build-essential
- tesseract-ocr
- git, curl, wget

🚀 Deployment Overview

Quick Overview of What Will Happen:

1. **Transfer** the deployment package to the VPS
2. **Extract** the files on the VPS
3. **Install** Node.js and system dependencies
4. **Configure** environment variables (API keys, URLs)
5. **Build** the Go backend
6. **Build** the Next.js frontend
7. **Create** systemd services for both applications
8. **Start** both services
9. **Configure** firewall for security
10. **Verify** everything works

Total Time: ~15-20 minutes

🔑 Configuration Required

Before deployment, you'll need:

Backend Configuration (`.env` file):

```
DB_PATH=./data/trading.db
UPLOAD_DIR=./data/uploads
PORT=8080
OPENALGO_URL=https://openalgo.mywire.org
OPENALGO_API_KEY=your_key_here          # ⚠️ YOU NEED THIS
ABACUS_API_KEY=your_key_here            # ⚠️ YOU NEED THIS
JWT_SECRET=random_secure_string          # ⚠️ CHANGE THIS
ENVIRONMENT=production
```

Frontend Configuration (`.env.local` file):

```
NEXT_PUBLIC_API_URL=http://67.211.219.94:8080
NEXT_PUBLIC_WS_URL=ws://67.211.219.94:8080/ws
```

🔒 Security Considerations

Default Credentials:

- Username: `admin`
- Password: `admin123`

⚠️ **CHANGE THESE IMMEDIATELY AFTER DEPLOYMENT!**

Ports Exposed:

- `3000` - Frontend web interface

- `8080` - Backend API

Firewall:

- Port 22 (SSH) - Required for access
- Port 3000 (Frontend) - Public access
- Port 8080 (Backend API) - Public access

📊 Post-Deployment Access

Once deployed, access your application at:

🌐 Web Interface:

```
http://67.211.219.94:3000
```

🔌 API Endpoint:

```
http://67.211.219.94:8080
```

🔍 Health Check:

```
http://67.211.219.94:8080/health
```

📖 Documentation Files

All available in the `trading-app` directory:

1. **README.md** - Application overview and features
2. **DEPLOYMENT.md** - Detailed deployment documentation
3. **QUICKSTART.md** - Quick setup guide
4. **deployment_guide.md** - VPS-specific step-by-step guide (NEW)

🛠️ Management Scripts

Pre-included scripts for easy management:

Script	Purpose	Usage
`deploy.sh`	Full automated deployment	`./deploy.sh`
`start.sh`	Start both services	`./start.sh`
`stop.sh`	Stop both services	`./stop.sh`
`restart.sh`	Restart both services	`./restart.sh`
`status.sh`	Check service status	`./status.sh`
`logs.sh`	View service logs	`./logs.sh backend` or `./logs.sh frontend`

✅ Verification Checklist

After deployment, verify:

- [ ] Backend service running (`systemctl status trading-backend`)
- [ ] Frontend service running (`systemctl status trading-frontend`)
- [ ] Port 8080 listening (`netstat -tulpn | grep 8080`)
- [ ] Port 3000 listening (`netstat -tulpn | grep 3000`)
- [ ] API health check responds (`curl http://localhost:8080/health`)
- [ ] Frontend accessible in browser (`http://67.211.219.94:3000`)
- [ ] Can log in with default credentials
- [ ] Firewall enabled with correct rules (`ufw status`)

🆘 Support Resources

If You Run Into Issues:

1. **Check the detailed deployment guide**: `/home/ubuntu/code_artifacts/deployment_guide.md`
2. **Review logs**:
   - Backend: `sudo tail -f /var/log/trading-backend.log`
   - Frontend: `sudo tail -f /var/log/trading-frontend.log`
3. **Check service status**: `sudo systemctl status trading-backend trading-frontend`
4. **Review README**: `/root/trading-app/README.md` (after extraction)

Common Issues & Solutions:

Port already in use:

```
sudo lsof -i :8080  # or :3000
sudo kill -9 [PID]
```

Service won't start:

```
sudo journalctl -u trading-backend -n 50
```

Out of memory:

```
# Add 2GB swap
sudo fallocate -l 2G /swapfile
sudo chmod 600 /swapfile
sudo mkswap /swapfile
sudo swapon /swapfile
```

📦 Package Contents

The `trading-app-deployment.tar.gz` includes:

✅ Complete Go backend source code
✅ Complete Next.js frontend source code
✅ All configuration scripts
✅ Deployment automation scripts
✅ Documentation files
✅ README and guides

❌ Does NOT include (will be built on VPS):
- node_modules (will be installed)
- .next build folder (will be built)
- Go binary (will be compiled)
- Database file (will be created)

🎯 Next Steps

1. **Read the deployment guide**: Open `deployment_guide.md`
2. **Prepare your API keys**:
   - Get your OpenAlgo API key
   - Get your Abacus.AI API key
3. **Transfer the package**: Use the scp command from the guide
4. **Follow the step-by-step instructions**: Each step is numbered and explained
5. **Verify deployment**: Use the checklist to confirm everything works

---

📞 Quick Contact

VPS IP: 67.211.219.94
SSH User: root
Frontend Port: 3000
Backend Port: 8080

Ready to Deploy? Start with the `deployment_guide.md`! 🚀