

Complete Render.com Deployment Guide

IU Quiz Application - React + FastAPI + PostgreSQL

Table of Contents

1. Hosting Plan Overview
 2. Pricing Breakdown
 3. Pre-Deployment Checklist
 4. Step-by-Step Deployment
 5. Post-Deployment Configuration
 6. Monitoring & Maintenance
 7. Troubleshooting
-

Hosting Plan Overview

Recommended Plan: Standard Tier

Perfect for 500-1000 concurrent users

Component	Service Type	Plan	Monthly Cost
Frontend (React + Vite)	Static Site	Free	\$0
Backend (FastAPI)	Web Service	Standard	\$25
Database (PostgreSQL)	Managed DB	Standard	\$25
TOTAL			\$50/month

What You Get:

- 2GB RAM for backend
 - 1 CPU core with auto-scaling
 - 10GB PostgreSQL database
 - 97 concurrent database connections
 - Unlimited bandwidth
 - Free SSL certificates
 - Global CDN
 - Auto-deploy from GitHub
 - 99.9% uptime SLA
 - Automatic daily backups
-

Pricing Comparison

Option 1: Starter (Testing/Low Traffic)

Frontend: \$0/month (Static Site - Free)
Backend: \$7/month (Starter - 512MB RAM)
Database: \$7/month (Essential - 1GB)

TOTAL: \$14/month

Good for: Up to 200 concurrent users, testing, MVP

Option 2: Standard (RECOMMENDED)

Frontend: \$0/month (Static Site - Free)
Backend: \$25/month (Standard - 2GB RAM)
Database: \$25/month (Standard - 10GB)

TOTAL: \$50/month

Good for: 500-1000 concurrent users, production

Option 3: Pro (High Traffic)

Frontend: \$0/month (Static Site - Free)
Backend: \$85/month (Pro - 4GB RAM)
Database: \$90/month (Pro - 50GB)

TOTAL: \$175/month

Good for: 1000+ concurrent users, enterprise

Payment Information

Billing Options:

1. **Monthly Billing:** Pay-as-you-go, cancel anytime
2. **Annual Billing:** Get 2 months FREE (save \$100/year)

Accepted Payment Methods:

- Credit/Debit Cards (Visa, Mastercard, Amex, Discover)
- PayPal

Annual Savings:

Monthly: $\$50 \times 12 = \$600/\text{year}$

Annual: $\$50 \times 10 = \$500/\text{year}$

SAVINGS: $\$100/\text{year}$ (16.7% discount)

Pre-Deployment Checklist

1. Code Preparation

- ☐ Push your code to GitHub
- ☐ Ensure `requirements.txt` exists in backend folder
- ☐ Ensure `package.json` exists in frontend folder
- ☐ Test locally (both frontend and backend working)

2. Required Files

Backend (/backend folder):

```
backend/  
  app/  
    main.py  
    database.py  
    models.py  
    crud.py  
    schemas.py  
    routers/  
requirements.txt  
.env (will be set in Render dashboard)
```

Frontend (/frontend folder):

```
frontend/  
  src/  
  public/  
package.json  
vite.config.js  
.env (will be set in Render dashboard)
```

3. GitHub Repository

- ☐ Create GitHub account (if you don't have one)
- ☐ Create new repository: `iu-quiz-app`
- ☐ Push your code:

```
cd "c:\IU Quiz"
git init
git add .
git commit -m "Initial commit"
git branch -M main
git remote add origin https://github.com/YOUR_USERNAME/iu-quiz-app.git
git push -u origin main
```

Step-by-Step Deployment

STEP 1: Create Render Account

1. **Go to:** <https://render.com>
2. **Click:** “Get Started” button (top right)
3. **Sign up with GitHub** (recommended):
 - Click “Sign up with GitHub”
 - Authorize Render to access your repositories
 - Complete profile setup
4. **Verify email** (check your inbox)

Tip: Using GitHub signup enables auto-deployment!

STEP 2: Create PostgreSQL Database

1. **From Dashboard:** Click “New +” button → Select “PostgreSQL”
2. **Configure Database:**

Name:	iu-quiz-db
Database:	indira_brainstorm
User:	(auto-generated)
Region:	Singapore (or closest to your users)
PostgreSQL Ver:	16 (latest)
Plan:	Standard (\$25/month)
3. **Click:** “Create Database”
4. **Wait:** 2-3 minutes for database to provision
5. **Copy Connection Details:**
 - Go to database dashboard
 - Find “Connections” section
 - Copy **Internal Database URL** (starts with `postgresql://`)
 - Save this for Step 3!

Example Internal URL:

postgresql://iu_quiz_db_user:xxxxx@dpg-xxxxx-a.singapore-postgres.render.com/indira_brainsto

STEP 3: Deploy Backend (FastAPI)

1. **From Dashboard:** Click “New +” → Select “Web Service”

2. **Connect Repository:**

- Select “Build and deploy from a Git repository”
- Click “Connect” next to your `iu-quiz-app` repository
- If not listed, click “Configure account” and grant access

3. **Configure Service:**

Name:	iu-quiz-backend
Region:	Singapore (same as database)
Branch:	main
Root Directory:	backend
Runtime:	Python 3
Build Command:	<code>pip install -r requirements.txt</code>
Start Command:	<code>uvicorn app.main:app --host 0.0.0.0 --port \$PORT</code>
Plan:	Standard (\$25/month)

4. **Add Environment Variables:** Click “Advanced” → “Add Environment Variable”

Key	Value
DATABASE_URL	(Paste Internal Database URL from Step 2)
CORS_ORIGINS	<code>https://YOUR-FRONTEND-URL.onrender.com</code> (will update later)
PYTHON_VERSION	3.11.0

5. **Click:** “Create Web Service”

6. **Wait:** 5-10 minutes for first deployment

7. **Verify Deployment:**

- Check logs for “Application startup complete”
 - Copy your backend URL: `https://iu-quiz-backend.onrender.com`
 - Test: Open `https://iu-quiz-backend.onrender.com/health`
 - Should see: `{"status":"healthy"}`
-

STEP 4: Deploy Frontend (React + Vite)

1. **From Dashboard:** Click “New +” → Select “Static Site”
2. **Connect Repository:**
 - Select your `iu-quiz-app` repository
 - Click “Connect”
3. **Configure Static Site:**

Name:	<code>iu-quiz-frontend</code>
Branch:	<code>main</code>
Root Directory:	<code>frontend</code>
Build Command:	<code>npm install && npm run build</code>
Publish Directory:	<code>dist</code>
4. **Add Environment Variables:** Click “Advanced” → “Add Environment Variable”

Key	Value
<code>VITE_API_URL</code>	<code>https://iu-quiz-backend.onrender.com</code>
5. **Click:** “Create Static Site”
6. **Wait:** 3-5 minutes for build and deployment
7. **Get Your URL:**
 - Your site will be at: `https://iu-quiz-frontend.onrender.com`
 - Copy this URL!

STEP 5: Update Backend CORS Settings

1. **Go to:** Backend service dashboard
2. **Click:** “Environment” tab
3. **Edit** `CORS_ORIGINS` variable:
`https://iu-quiz-frontend.onrender.com`
4. **Click:** “Save Changes”
5. **Wait:** Backend will auto-redeploy (2-3 minutes)

STEP 6: Initialize Database Tables

Option A: Using Render Shell

1. Go to backend service dashboard
2. Click “Shell” tab (top right)
3. Run:

```
python -c "from app.database import engine, Base; Base.metadata.create_all(bind=engine)"
```

Option B: Using Local Script

1. Update your local `.env` with production database URL
2. Run:

```
cd backend
python -c "from app.database import engine, Base; Base.metadata.create_all(bind=engine)"
```

STEP 7: Test Your Deployment

1. **Open Frontend:** <https://iu-quiz-frontend.onrender.com>
 2. **Test Registration:**
 - Fill in registration form
 - Click “Begin Quiz”
 - Should navigate to quiz page
 3. **Test Quiz:**
 - Answer questions
 - Submit quiz
 - Check results page
 4. **Test Admin Panel:**
 - Navigate to: <https://iu-quiz-frontend.onrender.com/admin>
 - Check leaderboard
 - Download PDF report
-

Post-Deployment Configuration

Custom Domain (Optional)

1. **Purchase Domain:** (e.g., from Namecheap, GoDaddy)
2. **In Render Dashboard:**
 - Go to Static Site settings
 - Click “Custom Domains”
 - Click “Add Custom Domain”
 - Enter: `quizapp.yourdomain.com`

3. **Update DNS Records** (at your domain registrar):

Type: CNAME
Name: quizapp
Value: iu-quiz-frontend.onrender.com
TTL: 3600

4. **Wait:** 10-60 minutes for DNS propagation

5. **SSL:** Automatically provisioned by Render

Environment Variables Reference

Backend (iu-quiz-backend):

`DATABASE_URL=postgresql://user:pass@host/db`
`CORS_ORIGINS=https://iu-quiz-frontend.onrender.com`
`PYTHON_VERSION=3.11.0`

Frontend (iu-quiz-frontend):

`VITE_API_URL=https://iu-quiz-backend.onrender.com`

Monitoring & Maintenance

Dashboard Metrics

Backend Service: - CPU Usage - Memory Usage - Request Rate - Response Time - Error Rate

Database: - Storage Used - Connection Count - Query Performance - Backup Status

Auto-Scaling

Render automatically scales your backend based on: - CPU usage > 80% - Memory usage > 80% - Request queue length

Backups

Automatic Daily Backups: - Retention: 7 days (Standard plan) - Time: 2:00 AM UTC - Location: Same region as database

Manual Backup: 1. Go to database dashboard 2. Click “Backups” tab 3. Click “Create Backup”

Logs

View Logs: 1. Go to service dashboard 2. Click “Logs” tab 3. Filter by: - Time range - Log level (info, error, warning) - Search term

Download Logs: - Click “Download” button - Exports last 1000 lines

Troubleshooting

Issue 1: Backend Not Starting

Symptoms: “Build failed” or “Service unhealthy”

Solutions: 1. Check `requirements.txt` is in `backend/` folder 2. Verify **Start Command:** `uvicorn app.main:app --host 0.0.0.0 --port $PORT` 3. Check logs for Python errors 4. Ensure `DATABASE_URL` environment variable is set

Issue 2: Database Connection Failed

Symptoms: “Connection refused” or “Could not connect to database”

Solutions: 1. Verify `DATABASE_URL` is the **Internal Database URL** 2. Check database is in same region as backend 3. Ensure database is running (check dashboard) 4. Test connection in Render Shell: `bash python -c "from app.database import engine; engine.connect()"`

Issue 3: Frontend Shows Network Error

Symptoms: Registration fails with “Network Error”

Solutions: 1. Check `VITE_API_URL` environment variable 2. Verify backend URL is correct (with `https://`) 3. Check CORS settings in backend 4. Hard refresh browser: `Ctrl + Shift + R` 5. Check browser console for errors

Issue 4: CORS Error

Symptoms: “Access-Control-Allow-Origin” error in browser console

Solutions: 1. Update backend `CORS_ORIGINS` to include frontend URL 2. Ensure no trailing slash in URLs 3. Redeploy backend after changing CORS settings 4. Clear browser cache

Issue 5: Build Failed

Frontend Build Errors: 1. Check `package.json` exists in `frontend/` folder 2. Verify **Build Command:** `npm install && npm run build` 3. Ensure **Publish Directory:** `dist` 4. Check for TypeScript/ESLint errors in logs

Backend Build Errors: 1. Check `requirements.txt` syntax 2. Verify Python version compatibility 3. Check for missing dependencies 4. Review build logs for specific errors

Cost Optimization Tips

1. Start Small, Scale Up

Month 1-2: Starter Plan (\$14/month)
Month 3+: Standard Plan (\$50/month) - when traffic grows
Later: Pro Plan (\$175/month) - if needed

2. Use Annual Billing

- Save \$100/year on Standard plan
- Save \$350/year on Pro plan

3. Monitor Usage

- Check metrics weekly
- Downgrade if traffic is low
- Upgrade before hitting limits

4. Optimize Database

- Delete old test data
- Archive old quiz attempts
- Use database indexes

5. Frontend Optimization

- Enable Vite build optimization
 - Use lazy loading for routes
 - Compress images
-

Support & Resources

Render Support

- **Email:** support@render.com
- **Response Time:** 12-24 hours (Standard plan)
- **Documentation:** <https://render.com/docs>

Community

- **Discord:** <https://discord.gg/render>
- **Forum:** <https://community.render.com>
- **Status Page:** <https://status.render.com>

Your App URLs (After Deployment)

Frontend: <https://iu-quiz-frontend.onrender.com>
Backend: <https://iu-quiz-backend.onrender.com>
Admin: <https://iu-quiz-frontend.onrender.com/admin>
Database: (Internal only, not publicly accessible)

Quick Reference

Monthly Costs Summary

Plan	Frontend	Backend	Database	Total
Starter	\$0	\$7	\$7	\$14
Standard	\$0	\$25	\$25	\$50
Pro	\$0	\$85	\$90	\$175

Deployment Checklist

- ☐ Create Render account
 - ☐ Push code to GitHub
 - ☐ Create PostgreSQL database
 - ☐ Deploy backend service
 - ☐ Deploy frontend static site
 - ☐ Update CORS settings
 - ☐ Initialize database tables
 - ☐ Test all features
 - ☐ Set up custom domain (optional)
 - ☐ Configure monitoring alerts
-

Important Notes

Free Tier Limitations: Render offers a free tier, but it has significant limitations (services spin down after 15 minutes of inactivity). For production use with 500-1000 users, you **MUST** use paid plans.

Billing: You're charged monthly starting from the day you upgrade to a paid plan. No refunds for partial months.

Security: Always use environment variables for sensitive data. Never commit `.env` files to GitHub.

Monitoring: Set up alerts for high CPU/memory usage to avoid service disruptions.

Auto-Deploy: Every push to your `main` branch will trigger automatic deployment. Use feature branches for development.

Success Criteria

Your deployment is successful when:

- Frontend loads at your Render URL
- Users can register and take quiz
- Quiz results are saved to database
- Admin panel shows participant data
- PDF export works
- No CORS errors in browser console
- Backend health check returns `{"status": "healthy"}`
- All 3 services show “Live” status in Render dashboard

Congratulations! Your IU Quiz App is now live on Render.com!

For questions or issues, refer to the Troubleshooting section or contact Render support.

Last Updated: February 10, 2026 Document Version: 1.0