 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Capstone Project	Deployment and Operations	
Project : - Job Gujarat	Date: 24-09-2025	Enrolment No: 92310133019 & 92310133013

Deployment and Operations

Introduction

Getting Job Gujarat from our laptops to actual production was quite a journey. We started with big dreams of using AWS or Google Cloud, but reality hit when we looked at our budget (basically zero) and our timeline. This document covers how we actually got the platform live, what we're doing to keep it running, and our plan for making sure it doesn't fall apart when thousands of people start using it.

1. Live Deployment

➤ What We Actually Built

Let me be honest - we didn't have money for fancy hosting. But we found a setup that actually works pretty well:

- Frontend on Vercel (their free tier is surprisingly good)
- Backend API on Render (also free, though it has some quirks)
- Database on Supabase (PostgreSQL + file storage in one place)

We picked this combo after trying a bunch of different options. Initially we thought of hosting everything on a single VPS, but managing servers isn't our strong point. These platforms handle most of the DevOps stuff for us, which means we can focus on actually building features.

➤ Frontend Deployment – Vercel

Getting the React app on Vercel was the easiest part. We literally just connected our GitHub repo, and it deployed automatically. But then came the fun parts.


First issue - environment variables. We kept wondering why our API calls weren't working in production. Turns out we were using process.env wrong. Vercel needs NEXT_PUBLIC_ prefix for client-side variables, but we're using plain React, so we had to configure build-time variables differently. Took us half a day to figure that out.

The production URL is <https://jobgujarat.vercel.app>. Setting up the domain was straightforward, but then we hit CORS issues (more on that later).

What's nice about Vercel is that every push to main automatically deploys. What's not nice is when you accidentally push broken code at 2 AM and the site goes down. We learned to use preview deployments the hard way.

➤ Backend on Render

Render seemed perfect - free Node.js hosting with 750 hours per month. What they don't tell you upfront is that the free tier spins down after 15 minutes of inactivity. So the first request after

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Capstone Project	Deployment and Operations	
Project : - Job Gujarat	Date: 24-09-2025	Enrolment No: 92310133019 & 92310133013

idle time takes like 30 seconds. Users think the site is broken.

Our workaround? We set up a simple cron job using cron-job.org that pings our API every 14 minutes. It's not elegant, but it keeps the server warm. The API endpoint is <https://jobgujarat-api.onrender.com>.

The deployment process itself was tricky. Render builds from our repo, but we had issues with Prisma. The build command needed to be:

```
npm install && npx prisma generate && npx prisma migrate deploy
```

Figuring out the right order took multiple failed deployments. Also, Render's build logs are hard to read - they dump everything in one stream. We started adding console.log statements everywhere just to debug deployment issues.

➤ Database Setup – Supabase

Supabase gave us Postgres + file storage, which is exactly what we needed. The free tier gives 500MB database and 1GB storage. Should be enough for now.

Setting up was smooth, but then we realized our backend on Render couldn't connect to Supabase. The connection string Supabase provides uses connection pooling through port 6543, but Render's free tier had issues with this. We had to use the direct connection string (port 5432) instead, which means we can't have too many concurrent connections.

For file storage, we created two buckets:

- resumes - private bucket for CV uploads
- avatars - public bucket for profile pictures

The tricky part was handling file uploads from the frontend through our backend to Supabase. We couldn't upload directly from browser to Supabase (security issues), so files go frontend -> backend -> Supabase. Not ideal for performance, but it works.


2. Monitoring Strategy

What We're Actually Monitoring

We don't have budget for fancy monitoring tools like DataDog or New Relic. Here's what we use: **Vercel Analytics** - Free tier gives us basic metrics. We can see:

- How many people visit
- Which pages they hit
- Load times (sort of - it's not very detailed)
- Where traffic comes from

➤ Render Dashboard - Shows:

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Capstone Project	Deployment and Operations	
Project : - Job Gujarat	Date: 24-09-2025	Enrolment No: 92310133019 & 92310133013

- RAM and CPU usage (we're usually at 60% RAM)
- Request count
- Deploy history
- Logs (last 100 lines only on free tier, which is annoying)

➤ **Supabase Dashboard** - Gives us:

- Database size (currently at 47MB)
- Number of API requests
- Slow queries (though we haven't optimized these yet)

UptimeRobot - Free external monitoring. Checks if our sites are up every 5 minutes. Sends email if down. Simple but effective.

➤ **Our KPIs (Keeping it Real)**


We track three main things:

1. **Is the site up?**
 - Target: Stay online 99% of the time
 - Reality: We're at about 98.5% (had two outages last month)
 - Measured by UptimeRobot
2. **Is it fast enough?**
 - Target: Pages load under 3 seconds
 - Reality: Homepage loads in 2.1s, job listing page takes 3.5s (needs work)
 - Measured by Vercel Analytics and manual testing
3. **Are jobs getting posted and applications submitted?**
 - Target: 95% success rate for submissions
 - Reality: 93% (we lose some to timeout errors)
 - Measured by comparing form submissions to database entries

➤ **Logs and Debugging**

Debugging production issues is painful with our setup. Render only keeps 100 lines of logs on free tier. Vercel functions logs are better but scattered.

We built a simple error tracking system - when something fails, we save error details to a database table. It's not sophisticated, but at least we can see what's breaking. Found out last week that file uploads were failing for files over 4MB because we hadn't set the body parser limit correctly.

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Capstone Project	Deployment and Operations	
Project : - Job Gujarat	Date: 24-09-2025	Enrolment No: 92310133019 & 92310133013

3. Maintenance Plan

➤ Daily Stuff We Check

Every morning, someone checks:

- UptimeRobot - any downtime alerts?
- Error logs table - new errors?
- Supabase dashboard - database size still OK?
- Quick manual test - can we post a job and apply?

This takes about 10 minutes. We rotate who does this so nobody gets stuck with it.

➤ Weekly Tasks

Every Sunday:

- Clear test data from database (we create fake jobs for testing)
- Check npm audit for security issues
- Review server logs for any patterns
- Test the full flow from job posting to application

We tried to automate this but honestly, manual checking catches more issues.

➤ Monthly Maintenance


First Saturday of each month:

- Full backup of database (download SQL dump from Supabase)
- Update dependencies (carefully - we broke login once by updating Auth0 SDK)
- Review and optimize slow queries
- Check if we're close to any free tier limits

➤ The Update Process

We don't have a fancy CI/CD pipeline. Here's what actually happens:

1. Make changes locally
2. Test thoroughly (we learned this after breaking production twice)
3. Push to a staging branch
4. Test on Vercel preview URL
5. If it works, merge to main
6. Watch Vercel and Render redeploy
7. Pray nothing breaks
8. Test production immediately

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Capstone Project	Deployment and Operations	
Project : - Job Gujarat	Date: 24-09-2025	Enrolment No: 92310133019 & 92310133013

For database changes, we run migrations locally first, test, then run on production during low traffic (usually 2-3 AM).

➤ Backup Strategy

Supabase does daily backups, but we also manually backup weekly. We download the SQL dump and save it to Google Drive (we have a shared folder). It's not automated, but it works. For code, everything's in Git. We tag releases when we remember (which isn't always).

4. Challenges and Reality Checks

CORS Issues

This was our biggest headache. Frontend on Vercel calling backend on Render caused CORS errors. We tried everything - wildcards, specific origins, different packages. Finally got it working with:

```
cors({
  origin: process.env.FRONTEND_URL,
  credentials: true
})
```

Simple, but took us two days to get right.

The Cold Start Problem

Render free tier cold starts are brutal. First request after idle takes 20-30 seconds. Users think it's broken. Our cron job workaround helps, but sometimes it fails and we don't notice until someone complains.

We're considering moving to Railway or paying for Render, but budget is tight.

File Size Drama


Render has a 512MB slug size limit on free tier. Our node_modules was 600MB. We had to:

- Remove dev dependencies
- Delete unused packages
- Use npm ci instead of npm install
- Add .slugignore file

Still barely fitting. Every new package is a careful decision.

Database Connection Issues

Free Supabase only allows 20 concurrent connections. During peak times (usually evening when people apply for jobs), we hit this limit. We added connection pooling in our backend, but it's not perfect. Sometimes queries just timeout.

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Capstone Project	Deployment and Operations	
Project : - Job Gujarat	Date: 24-09-2025	Enrolment No: 92310133019 & 92310133013

5. Evidence of Deployment

Live URLs:

- Main site: <https://job-gujarat.vercel.app>
- API: <https://job-gujarat-backend.onrender.com>
- Health check: <https://jobgujarat-api.onrender.com/health>

Conclusion

Look, our deployment isn't perfect. We're using free tiers, working around limitations, and sometimes things break. But it works. Real people are using it to find real jobs. The biggest lesson? You don't need expensive infrastructure to build something useful. You need to understand your constraints and work creatively within them. Every time we hit a limitation, we found a workaround. Not always elegant, but functional.

As we grow, we'll need to upgrade. The free tiers won't handle thousands of users. But for now, for proving the concept and serving our initial users, this scrappy setup does the job. And honestly, debugging these issues taught us more about deployment than any course could.