Codelab Educare LMS - Complete Deployment Guide

This guide will walk you through setting up the Codelab Educare Learning Management System on your local computer or a hosting server. No prior experience required!

Table of Contents

System Requirements

Minimum Requirements

- **Operating System**: Windows 10+, macOS 10.15+, or Ubuntu 18.04+
- **RAM**: 4GB minimum, 8GB recommended
- **Storage**: 2GB free space
- **Internet**: Stable connection for downloads and updates

Software Prerequisites

You'll need to install these before starting:

1. **Node.js** (version 18 or higher)
    - Download from: https://nodejs.org/
    - Choose the "LTS" (Long Term Support) version
    - During installation, check "Add to PATH" option
2. **Git** (for downloading project files)
    - Download from: https://git-scm.com/
    - Use default settings during installation
3. **PostgreSQL Database** (version 14 or higher)
    - Download from: https://www.postgresql.org/download/
    - Remember the password you set during installation!

Getting the Project Files

Method 1: Download from Replit

1. Go to your Replit project
2. Click the three dots menu (⋯) in the top right
3. Select "Download as zip"
4. Extract the zip file to a folder like `C:\codelab-educare` (Windows) or `~/codelab-educare` (Mac/Linux)

Method 2: Clone with Git (Advanced)

```
git clone [YOUR_REPLIT_GIT_URL] codelab-educare
cd codelab-educare
```

Local Development Setup

Step 1: Open Terminal/Command Prompt

- **Windows**: Press `Windows + R`, type `cmd`, press Enter
- **Mac**: Press `Cmd + Space`, type `terminal`, press Enter
- **Linux**: Press `Ctrl + Alt + T`

Step 2: Navigate to Project Folder

```
# Windows
cd C:\codelab-educare

# Mac/Linux
cd ~/codelab-educare
```

Step 3: Install Dependencies

```
npm install
```

This will download all required packages. It may take 5-10 minutes.

Database Setup

Step 1: Create Database

1. Open pgAdmin (installed with PostgreSQL) or use command line:

```
-- Connect to PostgreSQL and run these commands:
CREATE DATABASE codelab_educare;
CREATE USER codelab_user WITH ENCRYPTED PASSWORD 'your_secure_password';
GRANT ALL PRIVILEGES ON DATABASE codelab_educare TO codelab_user;
```

Step 2: Database Connection String

Your database URL will look like:

```
postgresql://codelab_user:your_secure_password@localhost:5432/codelab_educare
```

Environment Configuration

Step 1: Create Environment File

Create a file named `.env` in the project root folder:

```
# Database Configuration
DATABASE_URL=postgresql://codelab_user:your_secure_password@localhost:5432/code
lab_educare

# Session Security
SESSION_SECRET=your-super-secret-session-key-here-make-it-long-and-random

# Paystack Payment Integration (Nigerian Payment Gateway)
PAYSTACK_SECRET_KEY=sk_test_your_paystack_secret_key
VITE_PAYSTACK_PUBLIC_KEY=pk_test_your_paystack_public_key

# Perplexity AI Integration (Optional)
PERPLEXITY_API_KEY=your_perplexity_api_key_here

# Application Settings
NODE_ENV=development
PORT=5000
```

Step 2: Get API Keys

Paystack Setup (For Nigerian Payments)

1. Go to https://paystack.com/
2. Create an account
3. Go to Settings → API Keys & Webhooks
4. Copy your Test Secret Key and Test Public Key
5. Add them to your `.env` file

Perplexity AI Setup (Optional - for AI features)

1. Go to https://www.perplexity.ai/
2. Sign up for an account
3. Get your API key from the dashboard
4. Add it to your `.env` file

Running the Application

Step 1: Database Migration

```
npm run db:push
```

This creates all necessary database tables.

Step 2: Start the Application

```
npm run dev
```

Step 3: Access the Application

Open your web browser and go to:

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

Demo User Accounts

Login with these pre-configured accounts:

Admin Account:

- Email: `ummi.lawal@codelabeducare.com`
- Password: `Password1234`

Mentor Accounts:

- Email: `israel.alabi@codelabeducare.com`
- Password: `Password1234`

Student Accounts:

- Email: `oyinkonsola.ojobo@codelabeducare.com`
- Password: `Password1234`

Production Deployment

Option 1: VPS/Cloud Server (DigitalOcean, Linode, AWS)

Prerequisites

- Ubuntu 20.04+ server
- Root access or sudo privileges
- Domain name (optional but recommended)

Step 1: Server Setup

```
# Update system
sudo apt update && sudo apt upgrade -y

# Install Node.js
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt-get install -y nodejs

# Install PostgreSQL
sudo apt install postgresql postgresql-contrib -y

# Install PM2 (Process Manager)
sudo npm install -g pm2

# Install Nginx (Web Server)
sudo apt install nginx -y
```

Step 2: Database Setup

```
# Switch to postgres user
sudo -u postgres psql

# Create database and user
CREATE DATABASE codelab_educare;
CREATE USER codelab_user WITH ENCRYPTED PASSWORD 'your_secure_password';
GRANT ALL PRIVILEGES ON DATABASE codelab_educare TO codelab_user;
\q
```

Step 3: Deploy Application

```
# Clone your project
git clone [YOUR_REPO_URL] /var/www/codelab-educare
cd /var/www/codelab-educare

# Install dependencies
npm install

# Build the application
npm run build

# Set up environment variables
sudo nano .env
# (Add your production environment variables)

# Run database migrations
npm run db:push

# Start with PM2
pm2 start npm --name "codelab-educare" -- run dev
pm2 startup
pm2 save
```

Step 4: Nginx Configuration

```
sudo nano /etc/nginx/sites-available/codelab-educare
```

Add this configuration:

```
server {
    listen 80;
    server_name your-domain.com www.your-domain.com;

    location / {
        proxy_pass http://localhost:5000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_cache_bypass $http_upgrade;
    }
}
```

```
# Enable the site
sudo ln -s /etc/nginx/sites-available/codelab-educare /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl restart nginx
```

Step 5: SSL Certificate (Recommended)

```
# Install Certbot
sudo apt install certbot python3-certbot-nginx -y

# Get SSL certificate
sudo certbot --nginx -d your-domain.com -d www.your-domain.com
```

Option 2: Shared Hosting (cPanel)

Prerequisites

- Node.js support (version 18+)
- PostgreSQL database access
- SSH access (preferred) or File Manager

Step 1: Upload Files

1. Compress your project folder into a .zip file
2. Upload via cPanel File Manager or FTP
3. Extract in your domain's public folder

Step 2: Install Dependencies

```
# Via SSH
cd /path/to/your/domain
npm install --production
```

Step 3: Database Setup

1. Create PostgreSQL database via cPanel
2. Note the connection details
3. Update your `.env` file with hosting database URL

Step 4: Configure Application

```
# Build the application
npm run build

# Set up environment variables
# Create .env file with production settings
```

Troubleshooting

Common Issues

"Command not found: npm"

Solution: Node.js not installed properly

- Reinstall Node.js from https://nodejs.org/
- Restart your terminal/command prompt

"Database connection failed"

Solution: Check your database configuration

1. Verify PostgreSQL is running: `sudo systemctl status postgresql`
2. Check your DATABASE_URL in `.env` file
3. Test connection: `psql -h localhost -U codelab_user -d codelab_educare`

"Port 5000 already in use"

Solution: Change the port

1. Edit `.env` file: `PORT=3001`
2. Or kill the process: `npx kill-port 5000`

"Permission denied"

Solution: Fix file permissions

```
# On Linux/Mac
sudo chown -R $USER:$USER /path/to/project
chmod -R 755 /path/to/project
```

"Module not found" errors

Solution: Reinstall dependencies

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

Getting Help

1. **Check the logs**: `pm2 logs` (production) or check terminal output (development)
2. **Browser Console**: Press F12 → Console tab to see client-side errors
3. **Database logs**: Check PostgreSQL logs for database issues

Performance Optimization

For Production

1. **Enable Gzip compression** in Nginx
2. **Set up CDN** for static assets
3. **Configure database connection pooling**
4. **Set up monitoring** with PM2 or similar tools
5. **Regular database backups**

Database Backup

```
# Create backup
pg_dump -h localhost -U codelab_user codelab_educare > backup.sql

# Restore backup
psql -h localhost -U codelab_user codelab_educare < backup.sql
```

Security Considerations

Production Security Checklist

- [ ] Change all default passwords
- [ ] Use strong, unique SESSION_SECRET
- [ ] Enable HTTPS/SSL
- [ ] Configure firewall (UFW on Ubuntu)
- [ ] Regular security updates
- [ ] Database access restrictions
- [ ] API rate limiting
- [ ] Regular backups

Environment Variables Security

Never commit `.env` files to version control. Always use:

- Environment variables on server
- Secrets management in cloud platforms
- Encrypted configuration files

Maintenance

Regular Updates

```
# Update dependencies
npm update

# Update system (Ubuntu)
sudo apt update && sudo apt upgrade

# Restart application
pm2 restart codelab-educare
```

Monitoring

- Check application logs regularly
- Monitor disk space and memory usage
- Set up automated backups
- Monitor database performance

Support

If you encounter issues not covered in this guide:

1. Check the application logs first
2. Verify all environment variables are set correctly
3. Ensure all required services are running
4. Test with demo user accounts first

Remember to always test changes in a development environment before applying to production!