Fiverr

- fiver-tc
- alexbaik
- hiteshpatel601

fiver-tc

Terms and Conditions – Open Source Software Deployment Services

Thank you for choosing my service for deploying open-source applications such as **osTicket, Nextcloud, etc.** By placing an order, you agree to the following terms and conditions:

1. Service Scope

- Deployment will be performed as per the selected package (e.g., Shared Hosting, Standalone VPS/Server, or Docker-based setup).
- This service **only covers installation and initial configuration** of the selected application.
- I am **not responsible** for application-level support, customizations, troubleshooting internal software bugs, or usage training post-deployment.

2. Prerequisites to Start the Order

To begin work and ensure a smooth and successful deployment, you must provide the following:

- **Root SSH access** to the server or relevant hosting control panel access.
- A **valid domain name** and either:
  - DNS already pointed to the server IP, or
  - **Domain registrar access** if DNS configuration is needed on your behalf.
- If you have a **custom or paid SSL certificate**, please provide the necessary certificate files (CRT, KEY, and CA-Bundle if available).
- The server must be **fresh/clean**, with no conflicting applications or services running unless discussed in advance.
- If your hosting is **shared** (non-root), please confirm that it supports all required technologies (PHP, MySQL, etc.) for the requested application.

3. Delivery & Acceptance

- Once deployment is complete, I will notify you for review.
- You are responsible for **checking and testing the deployment** before marking the order as complete.

- Post-completion support is **not included** unless explicitly purchased as an add-on or agreed upon in the gig extras.

4. Responsibility Disclaimer

- I am not liable for:
  - Issues related to your hosting provider.
  - Application updates or third-party plugin/module issues.
  - Data loss due to your hosting environment or mismanagement after delivery.
- Security and backup setups are **not included** unless specifically requested and included in the order.

5. Communication & Order Delays

- Orders may be **delayed** or **cancelled** if prerequisites are not provided within a reasonable time (e.g., 48 hours).
- Incomplete or incorrect information may result in additional charges or cancellation depending on the complexity involved.

6. Additional Services

- Any service beyond the defined gig (e.g., application tuning, advanced security hardening, or customization) may incur extra charges.

alexbaik

?PBX Voice Recording Backup & Web Access Plan

? Scope

- Automatically archive and transfer PBX call recordings from GCP-hosted Linux servers to an on-premises FTP server.
- Provide web-based, password-protected access to recordings for internal staff.

? Daily/Weekly Backup Workflow

Step 1: Prepare the PBX Nodes (GCP Linux Hosts)
- Confirm recordings are stored under structured directories like:
/var/lib/freeswitch/recordings/{hostname}/archive/{year}/{month}/
- Ensure correct file ownership (e.g. www-data) and read access.

Step 2: Create a Bash Script for Scheduled Archiving
- Script will:
- Compress the current day/week’s recordings into a dated .tar.gz archive.
- Upload the archive securely to the FTP/scp server.

Step 3: Automate with Cron
- Schedule the script at 12:01 AM weekly or daily, depending on retention needs:

? Web Access Setup (On-Prem FTP Server)

Step 4: Install IIS & Configure Web Root
- Set FTP destination path as IIS web root:
C:\inetpub\wwwroot\pbx-recordings
- Configure basic authentication.

Step 5: Implement Access Restrictions
- Restrict login to local subnet IPs (e.g. 192.168.x.x) via IIS IP Restrictions.

Step 6: Password-Protect Recording Directory
- Use IIS features or .htaccess equivalent to secure the /pbx-recordings folder.

Step 7: Simple Listing Page (Optional)
- Create a minimal HTML index or use a file-listing tool for browsing recordings:

□□ Security Checklist
- □ Use strong FTP/SCP credentials.

- Validate archive before upload.
- Enable firewall rules to limit access.
- Monitor backups and access logs.

hiteshpatel601

Server	OS	Provider
1	Windows	Hetzner
2	Linux - Ubuntu	Hetzner
3	Linux - Ubuntu	Digital Ocean
4	Linux - Ubuntu	IONOS
5	Linux - Ubuntu	IONOS
6	Windows	
7	Windows	Clouldy
8	Linux - Ubuntu	IONOS

Prerequisites to Begin

1. Server Access

- □ SSH access to Linux/Ubuntu servers
- □ RDP or admin access to Windows servers

2. Network & Firewall Rules (optional if no internet restrictions)

- □ Allow outbound traffic to New Relic endpoints (TCP 443)
- □ If needed, whitelist these New Relic domains/IP ranges:

  New Relic Network Requirements

3. Server Info to Share

- Operating System and version for each server (already Shared)
- Hostname and internal IP
- Role (web, DB, app server, etc.)
- Business priority (helps tailor alert thresholds)

4. Admin Account

- Create a New Relic account or invite us to yours
- Provide admin-level access for setup

?? Setup Plan

Step 1: Install New Relic Infrastructure Agents

- Linux/Ubuntu: install via shell script or package manager
- Windows: install via MSI or PowerShell

Step 2: Enable Server-Level Alerts

- Alerts for:

- Server down
- High CPU/RAM usage
- Low disk space
- Network issues
- Application crash (if applicable)
- Synthetic Monitoring

Step 3: Dashboard Creation

- Custom dashboard showing server health, online status, CPU, RAM, disk, and real-time analytics

Step 4: Notification Channels

- Set up alerts via:

- Email

Step 5: Optional Add-ons

- Synthetic monitors for uptime checks