SISG — User Guide & Programmer Manual

SISG — Spoorthy Innovations Secure Gateway

User Guide & Programmer Maintenance Manual

Version	1.1 (June 2026)
Product	Spoorthy Innovations Secure Gateway (SISG)
URL	https://spoorthy.net/sisg/
Repository	github.com/Spoorthy-Innovations/sisg
Prepared for	Client stakeholders & development team

Table of Contents

---

1. Introduction

1.1 Purpose

SISG (Spoorthy Innovations Secure Gateway) is a secure web portal that lets authorized staff monitor, secure, and administer four AWS EC2 servers from a single dashboard. It replaces ad-hoc SSH, AWS Console, and manual firewall changes with a controlled, audited interface.

1.2 Who should read this document

Audience	Sections
End users (developers, ops staff)	Sections 3–5 — login, dashboard, IP manager, SSH keys, monitoring
Administrators	Section 4 — user management, audit log, Cloud Watch, key manager
Programmers / maintainers	Sections 6–12 — code structure, config, APIs, deployment, security
Client stakeholders	Sections 1–2, 5 (summary), 11 — capabilities and security posture

1.3 Key capabilities

- Centralized login with account lockout and OTP password reset
- Per-user server access control (AWS1–AWS4)
- AWS Security Group IP management with one-click “add my IP”
- SSH key provisioning (shared and per-user Ed25519 keys)
- Real-time performance monitoring (CPU, RAM, disk, network)
- Security risk scanning (8 modules) with history
- Cloud Watch — local server reachability monitoring with email alerts
- Audit logging of security-relevant actions
- phpMyAdmin access gated by SISG session

---

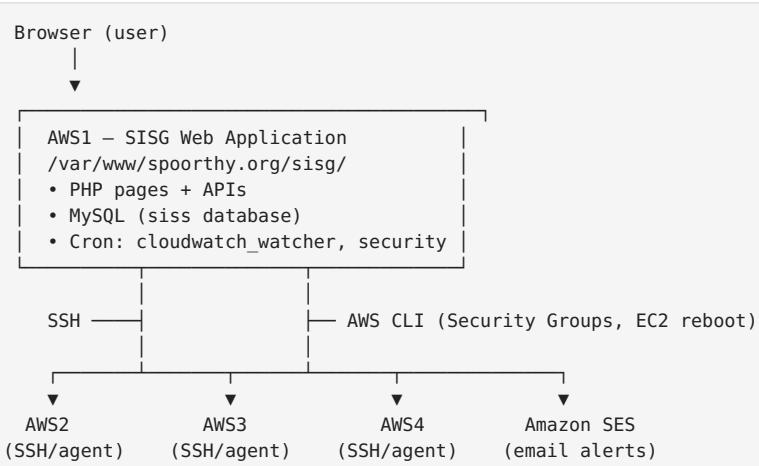
2. System Overview

2.1 Server landscape

SISG runs on **AWS1** and manages four servers in the **ap-south-1** (Mumbai) region:

ID	Public IP	Brand / domain	Primary role
AWS1	3.109.159.172	spoorthy.org / .net / .in	Web hosting, SISG portal, phpMyAdmin hub
AWS2	3.111.7.152	spoorthy.net, sujala.org	Development / Sujala applications
AWS3	43.204.27.215	hngpl.org	HNGPL dashboard and billing
AWS4	13.233.64.96	webapp.vaibhuinfratech.com	Vaibhu web application

2.2 How components connect



2.3 User roles

Role	Access
User	Dashboard modules filtered by `allowed_servers` (e.g. only AWS1, AWS2)
Admin	All servers + Audit Log + User Manager + SSH Key Manager + Cloud Watch configuration

---

3. User Guide

3.1 Logging in

1. Navigate to **https://spoorthy.net/sisg/login.php**
2. Enter your **User ID** (or mobile number) and **password**
3. Complete the invisible reCAPTCHA check (automatic)
4. On success you are redirected to the **Dashboard**

Session duration: 30 minutes of activity (secure cookie).

3.2 Forgot password

1. Click **Forgot Password** on the login page
2. Enter your User ID
3. Choose delivery method: **SMS** or **Email**
4. Enter the 6-digit OTP (valid 5 minutes)
5. Set a new password (minimum 6 characters)
6. Your account is reactivated if it was blocked

3.3 Dashboard

After login, the dashboard shows module tiles:

Tile	Description
phpMyAdmin	Opens database manager (only servers you are assigned)
IP Access Manager	Add or remove your IP address from firewall rules
SSH Access Keys	Download SSH private keys for server access
Performance Monitor	Live server metrics and health
Backup Manager	Placeholder for future backup features
Security Risk Analyzer	Run security scans on your servers
Cloud Watch	View server monitoring status and events
Audit Log	Admin only — security event history
User Manager	Admin only — manage accounts

Tiles marked **ADMIN** or with a lock icon are restricted to administrators.

3.4 Changing your password

While logged in, use **Change Password** from the profile menu (top-right) or navigate to `change_password.php`.

3.5 Theme

Click the sun/moon icon in the navigation bar to switch between **light** and **dark** mode. Your preference is saved in the browser.

---

4. Administrator Guide

4.1 User Manager

Path: Dashboard → User Manager (`user_manager.php`)

Action	How
Create user	Fill form: User ID, name, email, mobile, role, password
Assign servers	Check AWS1–AWS4 boxes under “Allowed Servers”
Block / activate	Toggle status or reset failed login count
Reset password	Set new password for the user

Important: Users only see data for servers listed in their `allowed_servers` field.

4.2 Audit Log

Path: Dashboard → Audit Log (`audit_log.php`)

Records: logins, failed attempts, account blocks, OTP events, IP changes, SSH key operations.

Unlock account: Find the blocked user and click **Unlock** to reset `pwdRetryCount` and set status to Active.

4.3 SSH Key Manager

Path: `manage_keys.php` (linked from admin navigation)

Matrix view: users × servers (AWS1–AWS4).

Action	Effect
Generate	Creates Ed25519 key pair, deploys public key to server
Revoke	Removes public key from server, marks key revoked in DB
Redeploy	Re-pushes a failed deployment

After bulk key changes, run `scripts/redeploy_sisg_keys.sh` on AWS1 to sync all keys.

4.4 Cloud Watch configuration

Path: Dashboard → Cloud Watch (`cloudwatch.php`) — **Admin configuration section**

SISG Cloud Watch is **not** AWS CloudWatch (to avoid service charges). It is a local watcher that:

- Checks TCP ports **22, 80, 443** on each server every minute
- Sends email when a server is unreachable for N minutes (default: 10)
- Optionally **auto-reboots** the EC2 instance within a time window (e.g. 22:00–08:00)

Setting	Meaning
Monitoring enabled	Turn checks on/off per server
Auto-reboot enabled	Allow automatic EC2 reboot when down
Down minutes	How long before alert/reboot
Window start/end	Hours when auto-reboot is permitted

Cron required: `* * * * * php /var/www/spoorthy.org/sisg/cloudwatch_watcher.php`

---

5. Module Reference

5.1 IP Access Manager

Path: `manage_ip.php`

Allows adding your current IP (or a custom IP) to AWS Security Groups for SSH, MySQL, or custom ports.

Adding an IP

1. The form pre-fills **your current IP**
2. Select **ports** (22 SSH, 3306 MySQL, or add custom port)
3. Select **server(s)** — AWS1, AWS2, AWS3, AWS4
4. Click **Add My Current IP** or **Add Custom IP**

Viewing rules

Rules are shown in **tabs per server**:

- **Custom Rules** — user-added IPs (can be deleted)
- **Protected Rules** — infrastructure IPs (cannot be deleted)

Deleting rules

Method	Steps
Single delete	Click **Delete** on one row
Bulk delete	Check 2+ rules → **Delete Selected**
Delete all (one server)	**Delete All Custom Rules** button
Delete all (all servers)	**Delete All Custom Rules (All Servers)** in page header

Protected IPs (never deleted): `3.109.159.172/32`, `3.111.7.152/32`, `43.204.27.215/32`, `13.233.64.96/32`, `172.31.0.0/16`, `0.0.0.0/0`.

Behind the scenes

When IPs are added or removed, SISG also:

1. Updates AWS Security Group via AWS CLI
2. Syncs **iptables** on the server (ports 22, 3306, 33060)
3. Updates MySQL `root@host` grants where applicable
4. Logs the action in `ip_access_log`

5.2 SSH Access Keys

Path: `download_ssh_key.php`

User type	Keys available
Admin	Master mesh key (all servers) + per-user keys if generated
Regular user	Per-server keys for assigned servers only

Download uses a **one-time token** (15-minute validity) for security.

Key formats: OpenSSH (`.rsa`) and PuTTY (`.ppk`).

5.3 Performance Monitor

Path: `performance.php` (hub) + sub-pages

Page	Refresh	Metrics
Performance Hub	10s	Overview all servers
CPU Monitor	10s	Load, usage breakdown, history charts, top processes
RAM Monitor	10s	Memory, swap, top consumers
Disk Monitor	30s	Volume usage, inode usage, critical alerts (>85%)
Network Monitor	10s	Bandwidth, connections, listening ports
System Health	On demand	Reboots, OOM, kernel panics, application error logs

Historical data is stored in MySQL (`cpu_history`, `ram_history`, etc.).

5.4 Security Risk Analyzer

Path: `security_risk.php`

Eight scan modules per server:

1. Open Port Scanner
2. SSH Configuration Audit

3. Firewall Rules (iptables)
4. Packages & CVEs
5. SSL/TLS Certificates (certbot)
6. Brute Force Detection
7. File Permissions
8. MySQL Security

Results can be saved to history and compared over time. Scheduled scans run via `security_watch.php` (every 6 hours).

5.5 phpMyAdmin

Path: `/mysql` (external to sisg folder)

- Requires active SISG login session
- Shows only database servers you are assigned
- Logging out of phpMyAdmin also signs you out of SISG

5.6 Backup Manager

Path: `backup.php` — currently a placeholder ("coming soon").

---

6. Architecture for Developers

6.1 Technology stack

Layer	Technology
Web server	Apache 2.x
Language	PHP 7.4+ (no framework)
Database	MySQL 8.0 (`siss`)
Frontend	Vanilla JS, Chart.js 4.4.4, CSS custom properties
Cloud	AWS EC2, Security Groups, SES
Remote access	SSH (ubuntu user), AWS CLI v2

6.2 Request flow (authenticated page)

```
HTTP request
    → session_start()
    → include db/config.php
    → check $_SESSION['session']
    → (optional) filter by allowed_servers
    → include header.php
    → page content
    → include footer.php
```

6.3 Key PHP files

Entry points

File	Auth	Notes
`index.php`	No	Public landing page
`login.php`	No	Login, OTP, logout
`dashboard.php`	Yes	Post-login menu
`manage_ip.php`	Yes	POST actions + AJAX partial

Shared libraries

File	Purpose
`audit.php`	`logAuditEvent()`, action constants
`servers_config.php`	Server definitions, SSH helpers
`aws_security_groups.php`	AWS CLI SG/NACL helpers
`security_scan_lib.php`	8-module scan engine
`agent_client.php`	HTTP client to distributed agents
`email.php`	SES SMTP via PHPMailer
`recaptcha.php`	reCAPTCHA v3 verify

Cron / CLI scripts

File	Schedule	Purpose
`cloudwatch_watcher.php`	Every minute	TCP health checks, alerts, auto-reboot
`security_watch.php`	0,6,12,18 daily	Scheduled security scans + email
`ssh_unresponsive_watch.php`	Legacy	Superseded by cloudwatch watcher

6.4 Agent subsystem

Distributed agents run on each EC2 at `/var/www/sisg-agent/`:

```
sisg/agent/
├── deploy.sh          # Deploy to all servers, generate tokens
├── run.php            # Core scan logic
├── lib.php            # DB, auth, heartbeat
├── cron_watch.php     # Per-server cron entry
└── agent_config.local.php  # Per-server config (not in git)
```

Hub polls agents via `agent_registry.local.php` (URLs + `X-SISG-Agent-Token`).

Deploy: `sudo bash /var/www/spoorthy.org/sisg/agent/deploy.sh`

6.5 IP Manager POST actions

`action`	Handler behavior
`add_ip`	`authorize-security-group-ingress` per server/port
`delete_ip`	Single rule revoke + iptables sync
`delete_ip_bulk`	Batch revoke, one iptables batch per server
`delete_all_custom`	All non-protected rules on one SG
`delete_all_custom_all_servers`	Loop all `$SERVERS`, purge custom rules
`reconcile_iptables`	Manual sync SG → iptables → MySQL

Core functions in `manage_ip.php`:

- `deleteCustomRule()` — permission check, AWS revoke, audit log, iptables
- `batchSyncIptables()` — batched add/remove with single `iptables-save`
- `runShellWithTimeout()` — prevents hung AWS/SSH commands

6.6 Adding a new dashboard module

1. Create `new_module.php` with session check and `include 'header.php'`
2. Set `$pageTitle` and `$activePage` before header
3. Add nav link in `header.php` if needed
4. Add dashboard tile in `dashboard.php`
5. If server-specific, filter with `allowed_servers` pattern from existing pages
6. If API-backed, add endpoint under `api/` with same session check

---

7. Configuration & Setup

7.1 Files to create from templates

Template	Target	Required
`db/config.php.example`	`db/config.php`	Yes
`email_config.local.php.example`	`email_config.local.php`	Yes (OTP/alerts)
`aws_ec2_sg_config.local.php.example`	`aws_ec2_sg_config.local.php`	For EC2 reboot
`agent_registry.local.php.example`	`agent_registry.local.php`	After agent deploy
`security_watch_config.local.php.example`	`security_watch_config.local.php`	Optional
`agent/agent_config.local.php.example`	On each server	Via deploy.sh

7.2 AWS credentials

Create `aws_cli_credentials.local` (gitignored):

```
[root]
aws_access_key_id = AKIA...
aws_secret_access_key = ...
region = ap-south-1
```

IAM permissions needed:

- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:RevokeSecurityGroupIngress`
- `ec2:DescribeSecurityGroups`
- `ec2:RebootInstances` (for Cloud Watch / manual reboot)
- `ec2:DescribeInstances`

7.3 SSH keys on AWS1

Path	Purpose
`/var/www/.ssh/id_rsa_aws1` – `id_rsa_aws4`	Hub SSH to each server
`keys/deploy/aws1.rsa` – `aws4.rsa`	Deploy user public keys
`keys/users/*`	Per-user private keys (gitignored)

7.4 Apache / .htaccess

- `sisg/.htaccess` — may redirect TLDs to spoorthy.net/sisg
- `keys/.htaccess` — `Deny from all` (block web access to keys)

7.5 Git deploy key

On AWS1, git remote uses SSH alias `github.com-sisg`:

```
~/.ssh/config:
  Host github.com-sisg
    HostName github.com
    IdentityFile ~/.ssh/sisg_github_deploy

git remote: git@github.com-sisg:Spoorthy-Innovations/sisg.git
```

Deploy key is registered on the GitHub repo (read/write).

---

8. Database Reference

Database name: `siss`
Host: `localhost` (on AWS1; agents have local copy on each server)

8.1 Core tables

`user`

Column	Type	Description
`userId`	VARCHAR	Login username (unique)
`pwd`	VARCHAR	Password
`name`	VARCHAR	Display name
`email`	VARCHAR	OTP email delivery
`mobile`	VARCHAR	OTP SMS delivery
`role`	VARCHAR	`Admin` or `User`
`status`	VARCHAR	`Active` or `Blocked - Max login attempts`
`pwdRetryCount`	TINYINT	Failed login counter
`otp`, `otp_expiry`	VARCHAR, DATETIME	Password reset OTP
`allowed_servers`	VARCHAR	e.g. `AWS1,AWS2,AWS3,AWS4`
`keyToken`, `keyValidity`	VARCHAR, DATETIME	Legacy SSH download token

`audit_log`

Column	Description
`user_id`	Who performed the action
`action`	e.g. `LOGIN_SUCCESS`, `LOGIN_FAILED`, `ACCOUNT_BLOCKED`
`details`	Human-readable text
`ip_address`	Client IP
`created_at`	Timestamp

`ip_access_log`

Logs IP manager add/delete actions: IP, port, servers, user.

`user_ssh_keys`

Per-user Ed25519 keys: `user_id`, `server_tag`, `public_key`, `private_key_path`, `deployed`, `revoked`.

8.2 Monitoring tables

Table	Stores
`cpu_history`	CPU %, load averages, per-server snapshots
`ram_history`	Memory and swap usage
`disk_history`	Mount points, usage %
`network_history`	Interface RX/TX, connections
`security_scan_history`	Full scan JSON + risk summary
`cloudwatch_controls`	Per-server monitor/reboot settings
`ssh_unresponsive_state`	Current up/down state per server
`ssh_unresponsive_events`	Outage and recovery event log

8.3 Agent tables

Defined in `db/agent_schema.sql`:

- `sisg_agent_state` — last scan, risk grade
- `sisg_agent_scans` — historical payloads
- `sisg_agent_alerts` — email alert log
- `sisg_agent_heartbeat` — last seen timestamp

---

9. API Reference

9.1 Authentication

Hub APIs check:

```
        if (!isset($_SESSION['session']) || empty($_SESSION['session'])) {
            http_response_code(401);
            echo json_encode(['error' => 'Unauthorized']);
            exit;
        }
```

Agent APIs check header: `X-SISG-Agent-Token: <token from agent_registry.local.php>`

9.2 Hub endpoints

`GET api/server_stats.php?type=all|cpu|ram|disk|network`

Returns live metrics for allowed servers. Stores snapshots in history tables.

`GET api/cpu_stats.php?server=aws1|all`

Live CPU stats for one or all servers.

`GET api/cpu_history.php?period=1h|6h|24h|7d&server=aws1`

Historical CPU data for charts.

`GET api/system_health.php?server=aws1`

Collects reboot reason, OOM, panics, app logs via SSH.

`GET api/security_scan.php?server=aws1`

Runs 8-module scan. Uses agent if registry configured.

`GET/POST api/user_keys.php`

action	Method	Description
`generate`	POST	Create key for user+server
`revoke`	POST	Revoke one key
`revoke_all`	POST	Revoke all keys for user
`deploy`	POST	Re-deploy existing key
`status`	GET	Key deployment status
`list`	GET	All keys (admin)

`POST api/ec2_reboot.php`

Body: `server=aws1` — triggers EC2 reboot via AWS CLI.

9.3 Agent endpoints (on each server)

Endpoint	Auth	Purpose
`api/agent/ping.php`	No	Liveness check
`api/agent/scan.php`	Yes	Run security scan
`api/agent/status.php`	Yes	Last scan + heartbeat
`api/agent/alerts.php`	Yes	Local alert log

---

10. Deployment & Operations

10.1 Standard file deploy

```
# From development machine → AWS1
scp -i ~/.ssh/id_rsa_aws1 file.php ubuntu@3.109.159.172:/tmp/
ssh -i ~/.ssh/id_rsa_aws1 ubuntu@3.109.159.172 \
  "sudo cp /tmp/file.php /var/www/spoorthy.org/sisg/ && \
   sudo chown www-data:www-data /var/www/spoorthy.org/sisg/file.php"
```

10.2 Git workflow

```
cd /var/www/spoorthy.org/sisg
git pull origin main
# test changes
git add <files>
git commit -m "Description"
git push origin main
```

10.3 Agent deploy

```
cd /var/www/spoorthy.org/sisg
sudo bash agent/deploy.sh
```

This rsyncs the agent bundle, applies DB schema, and writes `agent_registry.local.php`.

10.4 SSH key redeploy

After user key changes:

```
sudo bash /var/www/spoorthy.org/sisg/scripts/redeploy_sisg_keys.sh
```

10.5 Cron jobs (AWS1 crontab)

```
* * * * * www-data php /var/www/spoorthy.org/sisg/cloudwatch_watcher.php >>
/var/www/spoorthy.org/sisg/logs/cloudwatch.log 2>&1
0 0,6,12,18 * * * www-data php /var/www/spoorthy.org/sisg/security_watch.php >>
/var/www/spoorthy.org/sisg/logs/security_watch.log 2>&1
```

10.6 Log locations

Path	Content
`sisg/logs/`	Application logs (gitignored)
`/var/log/apache2/`	Apache access/error
`audit_log` table	Security events (query via UI)

10.7 phpMyAdmin external files

Deploy separately to `/var/www/spoorthy.org/mysql/`:

- `external/mysql/index.php` — session gate
- `external/mysql/config.inc.php` — 4-server config

---

11. Security

11.1 Controls in place

Control	Implementation
Login brute-force protection	3 attempts → account block + admin email
Bot protection	reCAPTCHA v3 on login and forgot password
Session security	Secure cookie, 30-minute timeout
Server access RBAC	`allowed_servers` column, enforced in UI and APIs
SSH key protection	`.htaccess` deny, one-time download tokens
Infrastructure IP protection	Cannot delete server/VPC/public CIDRs in IP manager
Host firewall sync	iptables DROP on MySQL ports beyond SG
Audit trail	All login, OTP, IP, and key events logged
Agent authentication	Shared secret token per server

11.2 Operator responsibilities

1. **Never commit** `db/config.php`, AWS credentials, or private keys
2. Restrict filesystem permissions: `keys/`, `aws_cli_credentials.local` → `www-data` only
3. Use IAM least-privilege for AWS CLI user
4. Rotate agent tokens after redeploy
5. Set strong user passwords; review Audit Log regularly
6. Review Cloud Watch auto-reboot settings before enabling

11.3 Known limitations (for maintainers)

Area	Note
Password storage	Plaintext in `user.pwd` — consider hashing migration
SQL queries	Mix of prepared statements and escaped strings
AWS CLI from web	`shell_exec` as www-data — high impact if RCE
SSH	`StrictHostKeyChecking=no` on automated connections

---

12. Troubleshooting

12.1 Cannot log in

Symptom	Check
“Account blocked”	Admin unlocks via Audit Log, or user resets password
reCAPTCHA error	Verify keys in `recaptcha.php`, domain registered in Google console
Wrong password	3 strikes blocks account

12.2 IP not working after add

1. Confirm rule appears in IP Manager custom rules tab
2. Check AWS CLI credentials: `aws sts get-caller-identity --profile root`

3. Run reconcile if iptables out of sync
4. Verify you added `/32` CIDR (SISG adds automatically)

12.3 SSH key download fails

1. Check `keys/users/` or deploy keys exist on disk
2. Verify user has `allowed_servers` including target
3. Check `ssh_key_tokens` table for expired/used tokens

12.4 Performance data empty

1. SSH from AWS1 to target: `ssh -i /var/www/.ssh/id_rsa_aws2 ubuntu@3.111.7.152`
2. Check `api/server_stats.php` returns JSON (browser dev tools)
3. Verify `allowed_servers` includes the server

12.5 Cloud Watch not alerting

1. Confirm cron is running: `grep cloudwatch /etc/cron*`
2. Check `cloudwatch_controls` table — `monitoring_enabled = 1`
3. Verify SES credentials in `email_config.local.php`
4. Review `logs/cloudwatch.log`

12.6 Git push fails

```
ssh -T git@github.com-sisg   # Should say: Hi Spoorthy-Innovations/sisg!
```

If permission denied, verify deploy key on GitHub repo settings.

---

13. Appendix

13.1 Audit action constants

Constant	Meaning
`LOGIN_SUCCESS`	Successful login
`LOGIN_FAILED`	Wrong password
`LOGIN_BLOCKED`	Login attempt on blocked account
`LOGOUT`	User signed out
`PASSWORD_RESET_REQUEST`	Forgot password started
`PASSWORD_RESET_SUCCESS`	Password changed
`OTP_SENT`	OTP dispatched
`OTP_VERIFIED`	Correct OTP entered
`OTP_FAILED`	Wrong/expired OTP
`ACCOUNT_UNLOCKED`	Admin unlocked account

13.2 Security Group IDs

Server	Security Group ID	Name
AWS1	sg-0638a6f9faf0eb614	spoorthy-aws1
AWS2	sg-0ec7a5500043b1648	sujala-aws2
AWS3	sg-0d689a2a50a4f54c7	hngpl-aws3
AWS4	sg-0586e7fa1c464df41	vaibhu-aws4

13.3 Common ports

Port	Service	Notes
22	SSH	Managed via IP Manager
80, 443	HTTP/S	Open globally (protected rules)
3306	MySQL	Restricted to server IPs + VPC
33060	MySQL X Protocol	Synced via iptables

13.4 Document revision history

Date	Version	Changes
2026-03	1.0	Initial platform documentation
2026-06	1.1	Cloud Watch, bulk IP delete, per-user SSH keys, agent subsystem, deploy key

---

Spoorthy Innovations Pvt Ltd
https://spoorthy.org