Robokeys ShellGuard: AI Command Approval & Terminal Automation

Welcome to the Robokeys ShellGuard User Guide. This guide is designed for **DevOps engineers and developers**, helping you securely integrate AI-driven automation into terminal workflows using **ShellGuard**.

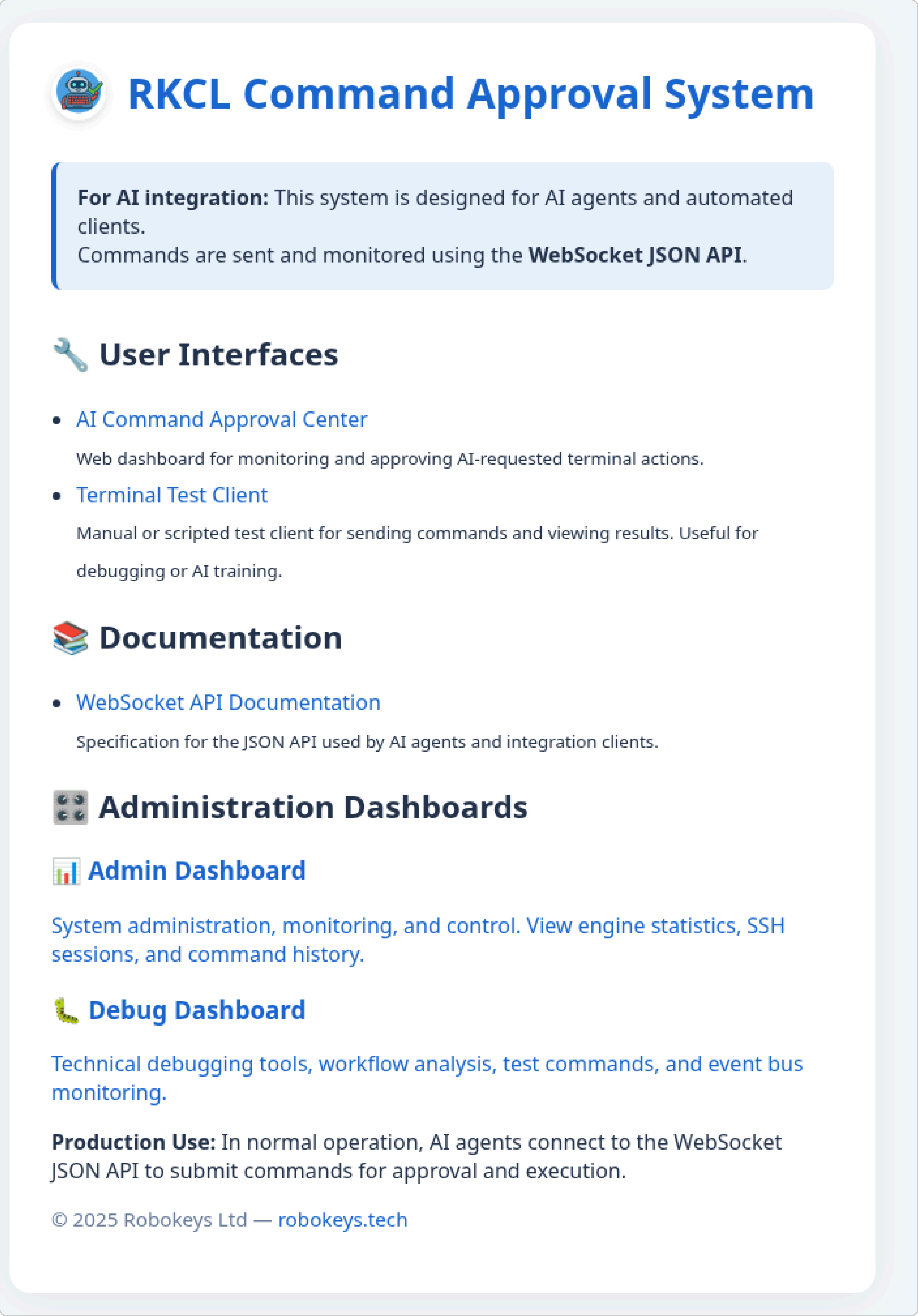
> **Key Features:** Secure AI command execution, human approval workflows, WebSocket API for AI integration, full audit trail.

1. Introduction

Robokeys ShellGuard provides a secure and auditable bridge between AI agents and terminal environments. It ensures that commands executed by automated systems are vetted and controlled, reducing risk and maintaining compliance.

- **Enhanced Security:** AI-driven commands can require human approval.
- **Full Auditing:** Logs every command, approval, and outcome.
- **Controlled Automation:** Executes commands safely within defined boundaries.
- **Developer Efficiency:** Built-in tools for testing, debugging, and training AI agents.

2. System Overview



RKCL Command Approval System

> **For AI integration:** This system is designed for AI agents and automated clients.
> Commands are sent and monitored using the **WebSocket JSON API**.

🔧 User Interfaces

- AI Command Approval Center

  Web dashboard for monitoring and approving AI-requested terminal actions.
- Terminal Test Client

  Manual or scripted test client for sending commands and viewing results. Useful for debugging or AI training.

📚 Documentation

- WebSocket API Documentation

  Specification for the JSON API used by AI agents and integration clients.

🎛️ Administration Dashboards

📊 Admin Dashboard

System administration, monitoring, and control. View engine statistics, SSH sessions, and command history.

🐛 Debug Dashboard

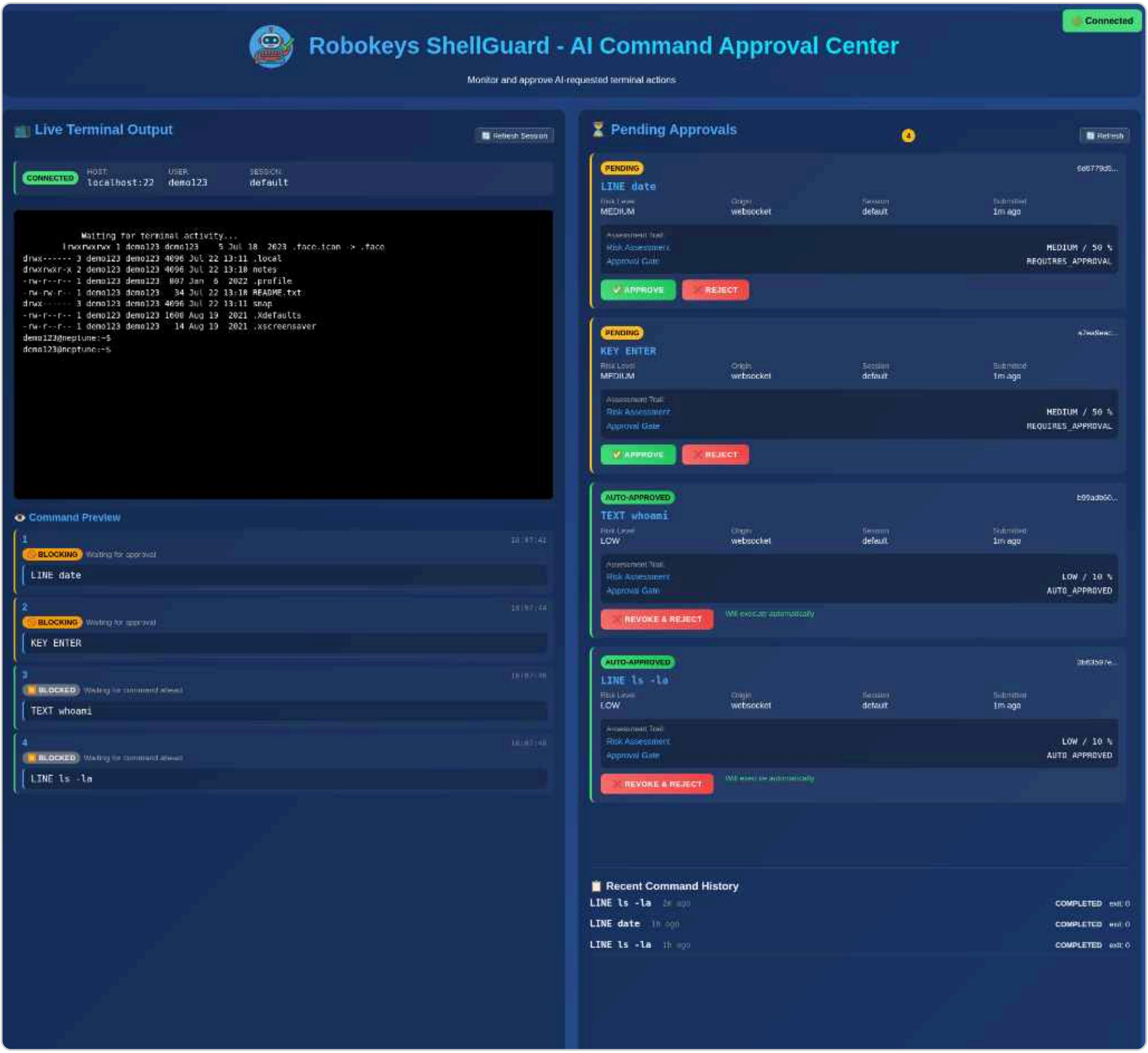
Technical debugging tools, workflow analysis, test commands, and event bus monitoring.

Production Use: In normal operation, AI agents connect to the WebSocket JSON API to submit commands for approval and execution.

The system is built around the RKCL Command Approval Engine and includes these main components:

- **AI Command Approval Center:** Web dashboard for monitoring and approving actions.
- **Terminal Test Client:** For manual and automated test interactions.
- **Admin Dashboard:** SSH session monitoring and system stats.
- **Debug Dashboard:** For troubleshooting workflows and event activity.

3. AI Command Approval Center



HOST: localhost:22 USER: demo123 SESSION: default

```
          Waiting for terminal activity...
        lrwxrwxrwx 1 demo123 demo123    5 Jul 18  2023 .face.icon -> .face
drwx------ 3 demo123 demo123 4096 Jul 22 13:11 .local
drwxrwxr-x 2 demo123 demo123 4096 Jul 22 13:10 notes
-rw-r--r-- 1 demo123 demo123  807 Jan  6  2022 .profile
-rw-rw-r-- 1 demo123 demo123   34 Jul 22 13:18 README.txt
drwx------ 3 demo123 demo123 4096 Jul 22 13:11 snap
-rw-r--r-- 1 demo123 demo123 1600 Aug 19  2021 .Xdefaults
-rw-r--r-- 1 demo123 demo123   14 Aug 19  2021 .xscreensaver
demo123@neptune:~$
demo123@neptune:~$
```

Command Preview

1 — BLOCKING — Waiting for approval — 18:07:42
`LINE date`

2 — BLOCKING — Waiting for approval — 18:07:44
`KEY ENTER`

3 — BLOCKED — Waiting for command ahead — 18:07:46
`TEXT whoami`

4 — BLOCKED — Waiting for command ahead — 18:07:48
`LINE ls -la`

Pending Approvals

PENDING — 6d6779d0...
`LINE date`
Risk Level: MEDIUM | Origin: websocket | Session: default | Submitted: 1m ago
Assessment Trail
Risk Assessment: MEDIUM / 50 %
Approval Gate: REQUIRES_APPROVAL

PENDING — a7ea8eac...
`KEY ENTER`
Risk Level: MEDIUM | Origin: websocket | Session: default | Submitted: 1m ago
Assessment Trail
Risk Assessment: MEDIUM / 50 %
Approval Gate: REQUIRES_APPROVAL

AUTO-APPROVED — b99adb60...
`TEXT whoami`
Risk Level: LOW | Origin: websocket | Session: default | Submitted: 1m ago
Assessment Trail
Risk Assessment: LOW / 10 %
Approval Gate: AUTO_APPROVED
Will execute automatically

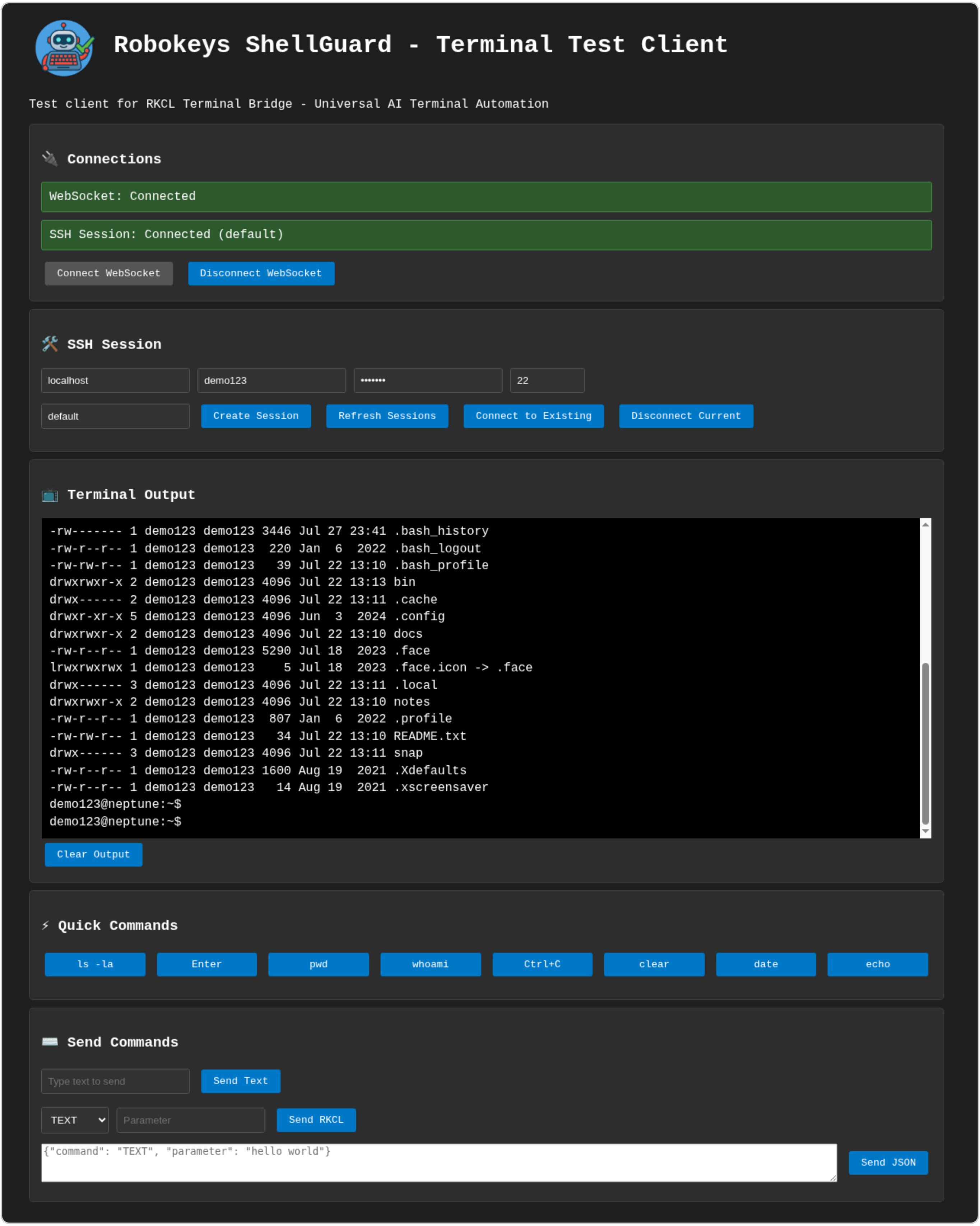
AUTO-APPROVED — 3b63597e...
`LINE ls -la`
Risk Level: LOW | Origin: websocket | Session: default | Submitted: 1m ago
Assessment Trail
Risk Assessment: LOW / 10 %
Approval Gate: AUTO_APPROVED
Will execute automatically

Recent Command History

```
LINE ls -la   2h ago    COMPLETED  exit: 0
LINE date     1h ago    COMPLETED  exit: 0
LINE ls -la   1h ago    COMPLETED  exit: 0
```

This dashboard displays pending approvals, auto-approved commands, and allows human intervention when needed.

4. Terminal Test Client



Robokeys ShellGuard - Terminal Test Client

Test client for RKCL Terminal Bridge - Universal AI Terminal Automation

🔌 Connections

WebSocket: Connected

SSH Session: Connected (default)

🛠 SSH Session

localhost | demo123 | ••••••• | 22

default

📺 Terminal Output

```
-rw------- 1 demo123 demo123 3446 Jul 27 23:41 .bash_history
-rw-r--r-- 1 demo123 demo123  220 Jan  6  2022 .bash_logout
-rw-rw-r-- 1 demo123 demo123   39 Jul 22 13:10 .bash_profile
drwxrwxr-x 2 demo123 demo123 4096 Jul 22 13:13 bin
drwx------ 2 demo123 demo123 4096 Jul 22 13:11 .cache
drwxr-xr-x 5 demo123 demo123 4096 Jun  3  2024 .config
drwxrwxr-x 2 demo123 demo123 4096 Jul 22 13:10 docs
-rw-r--r-- 1 demo123 demo123 5290 Jul 18  2023 .face
lrwxrwxrwx 1 demo123 demo123    5 Jul 18  2023 .face.icon -> .face
drwx------ 3 demo123 demo123 4096 Jul 22 13:11 .local
drwxrwxr-x 2 demo123 demo123 4096 Jul 22 13:10 notes
-rw-r--r-- 1 demo123 demo123  807 Jan  6  2022 .profile
-rw-rw-r-- 1 demo123 demo123   34 Jul 22 13:10 README.txt
drwx------ 3 demo123 demo123 4096 Jul 22 13:11 snap
-rw-r--r-- 1 demo123 demo123 1600 Aug 19  2021 .Xdefaults
-rw-r--r-- 1 demo123 demo123   14 Aug 19  2021 .xscreensaver
demo123@neptune:~$
demo123@neptune:~$
```

⚡ Quick Commands

⌨ Send Commands

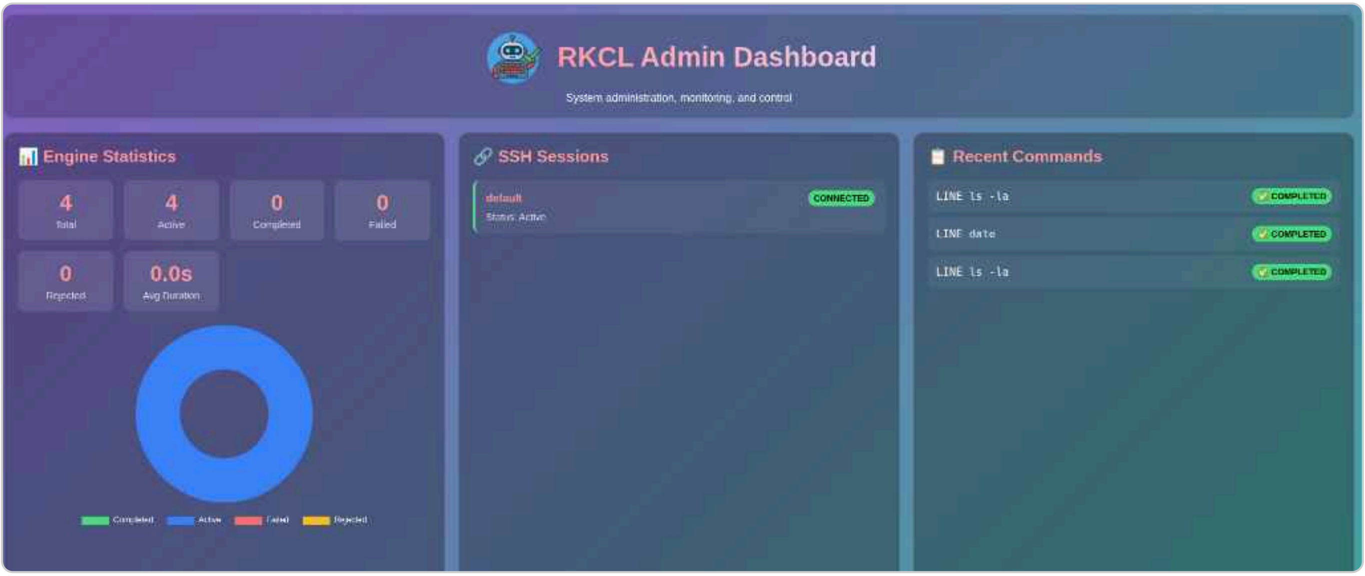
Type text to send

TEXT | Parameter

```
{"command": "TEXT", "parameter": "hello world"}
```

The Terminal Test Client lets developers send commands manually or script interactions for AI training and debugging. It connects via the same WebSocket interface that AI systems use, ensuring realistic tests.

5. Admin Dashboard



RKCL Admin Dashboard

System administration, monitoring, and control

Engine Statistics

4 Total	4 Active	0 Completed	0 Failed
0 Rejected	0.0s Avg Duration		

SSH Sessions

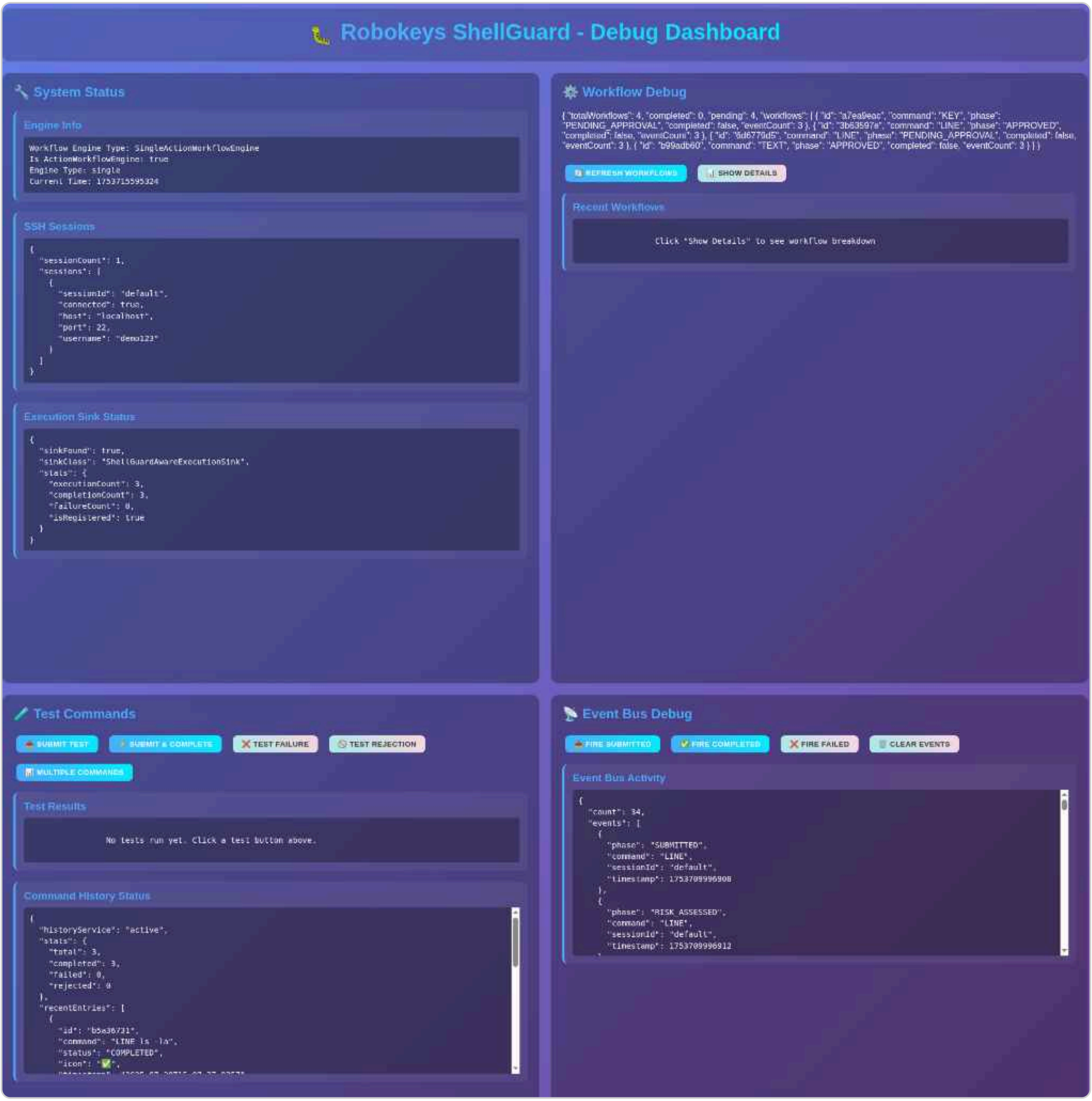
default CONNECTED
Status: Active

Recent Commands

```
LINE ls -la    ✓ COMPLETED
LINE date      ✓ COMPLETED
LINE ls -la    ✓ COMPLETED
```

Provides real-time visibility of SSH sessions, command execution status, and workflow performance.

6. Debug Dashboard



```
Workflow Engine Type: SingleActionWorkflowEngine
Is ActionWorkflowEngine: true
Engine Type: single
Current Time: 1753715595324
```

```
{
  "sessionCount": 1,
  "sessions": [
    {
      "sessionId": "default",
      "connected": true,
      "host": "localhost",
      "port": 22,
      "username": "demo123"
    }
  ]
}
```

```
{
  "sinkFound": true,
  "sinkClass": "ShellGuardAwareExecutionSink",
  "stats": {
    "executionCount": 3,
    "completionCount": 3,
    "failureCount": 0,
    "isRegistered": true
  }
}
```

```
{ "totalWorkflows": 4, "completed": 0, "pending": 4, "workflows": [ { "id": "a7ea9eac", "command": "KEY", "phase":
"PENDING_APPROVAL", "completed": false, "eventCount": 3 }, { "id": "3b63597e", "command": "LINE", "phase": "APPROVED",
"completed": false, "eventCount": 3 }, { "id": "5d6779d5", "command": "LINE", "phase": "PENDING_APPROVAL", "completed": false,
"eventCount": 3 }, { "id": "b99adb60", "command": "TEXT", "phase": "APPROVED", "completed": false, "eventCount": 3 } ] }
```

```
Click "Show Details" to see workflow breakdown
```

```
No tests run yet. Click a test button above.
```

```
{
  "historyService": "active",
  "stats": {
    "total": 3,
    "completed": 3,
    "failed": 0,
    "rejected": 0
  },
  "recentEntries": [
    {
      "id": "b5a36731",
      "command": "LINE ls -la",
      "status": "COMPLETED",
      "icon": "✅",
```

```
{
  "count": 34,
  "events": [
    {
      "phase": "SUBMITTED",
      "command": "LINE",
      "sessionId": "default",
      "timestamp": 1753709996908
    },
    {
      "phase": "RISK_ASSESSED",
      "command": "LINE",
      "sessionId": "default",
      "timestamp": 1753709996912
```

Offers deep insights into workflow state, event bus activity, and processing timelines for advanced troubleshooting.

7. WebSocket API

The primary integration point for AI agents is the **WebSocket JSON API**. It allows secure, structured communication with ShellGuard for command approval and execution.

Example Connection

Connect your client to the WebSocket endpoint (replace

```
yourhost
```

as needed):

```
ws://yourhost:8080/rkcl/ws
```

Example Request

```
{
  "command": "ls",
  "session": "default"
}
```

Example Response

```
{
  "status": "PENDING",
  "id": "cmd-12345",
  "message": "Command submitted for approval"
}
```

Once approved, the response will include execution output:

```
{
  "status": "APPROVED",
  "result": "file1.txt file2.txt"
}
```

> **Tip:** Use the Terminal Test Client to validate your integration before deploying AI agents.

8. Quickstart Guide

1. Start the ShellGuard server with your configuration.
2. Access the **Approval Center** at

   ```
   http://yourhost:8080/approval
   ```

   .
3. Use the **Terminal Test Client** to send test commands.
4. Integrate your AI system using the **WebSocket API**.