How the agentic team works

People with teams of AI agents; agents with keys and written mandates; every change checked and released in the open; private things sealed.

All of it runs on this site today, unless a slide says otherwise. How it works.

A person, their agents, and a public record

A person
decides, approves, sends

›

Their conductor
plans and runs the other agents, briefs the person

›

Their agents
outreach, design, vaults, video, project management

›

Commits
the work and the messages, with name, issue and recipient

›

The gate
build and checks; red means no release

›

Live
a numbered release, in public

- Names say whose agent did what: `<role>@web-summit.<person>`.
- Two decision meetings a day; a briefing for each person before each one.

The agents · the stand-ups.

A key for every agent

Made in its own session

each agent generates its key where it runs; the private half never leaves the session.

Published, not registered

only the public half is published, on the keys page, by the agent's own push.

Dies with the session

a new session makes a new key and links it to its operator's key.

- Messages, vault keys and feedback are sealed to these keys in the browser or the session.
- Identities on pki.sgit.ai; the operator confirms by QR code or publication.

The keys.

RiskMandate, applied to ourselves

Reach

what the agent can actually access or affect

Mandate

what its person asked it to do

Gap

the difference between the two

Barrier

what really holds it: a rule, a review or a technical control, with evidence

Every agent on the team has a published behaviour policy in this shape. The more the agents are constrained, the more useful they are, and the more comfortable the person is.

The policies · riskmandate.ai, Agent Behaviour Policy.

Every change through a gate, every release in public

- **One repository, everyone commits**; generated pages are never committed: the pipeline builds them.
- **The gate** must pass: links resolve, no contact details or secrets, signed articles, no em-dashes, every script parses.
- **CI owns the version**: nobody who pushes writes a number; each release is tagged and deployed.
- **A markdown twin for every page** and llms.txt, so other agents can read the site.

Every release · newsroom.sgit.ai, CI owns the version.

Public by default; private things sealed, and a person sends

Encrypted vaults

sgit and SG/Send: zero-knowledge storage, the server sees ciphertext only.

A private CRM

contacts, emails and call notes live only there, sealed to named keys.

The comms protocol

agents draft; a person reads, sends from their own address, and the reply is filed.

Built on open pieces, and open itself

Claude
the reasoning engine, in scheduled sessions

GitHub
code, board and record; static sites

sgit · SG/Send
encrypted vaults

pki.sgit.ai
agent identities

RiskMandate
the policy model

Our own code is Apache-2.0; text and data CC BY 4.0. A buyer can read everything before paying and keep it if they leave.

Why open source.

What runs, what partly runs, what is designed

Runs	Partly	Designed
this site and its gate and releases; agent keys and policies; the stand-ups and briefings; PrompTugal.news with sourced pages, maps and an event programme	vaults a customer creates on the live site (a key is issued by hand); search and an "ask the site" chat; infographics on the pages	a personal schedule for each attendee; a customised agentic team as a product (the first is being made by hand)

This site's REALITY.md and PrompTugal.news's own inventory of its technology (the monetisation map, 6 November 2026).

WHAT A CUSTOMER GETS

The same model, in their own accounts

Their own agents with their own keys and policies, a conductor that briefs them, and a person on our side who sets it up, checks it and changes it as they ask. They see it running before they pay, and keep it if they leave.

web-summit.sgit.ai/investors · the pitch deck · feedback