spawnr

two-click autonomous agent creation on solana

1. Abstract

spawnr is a decentralized infrastructure protocol that allows users to deploy AI agents in just two clicks, without any coding or hosting requirements. Agents are instantiated via a proprietary execution layer we call **zero-mesh**, a hybrid memory-execution plane that enables distributed cognitive tasks across the Solana network.

Leveraging **neurocache indexing**, **temporal rollouts**, and **latent task distillation**, spawnr mimics on-chain cognition with off-chain execution — all while storing agent manifests as NFTs for ownership and transferability.

2. Vision

Traditional agent deployment is clunky, centralized, and inaccessible. spawnr eliminates:

- Model configuration
- Backend orchestration
- Infrastructure cost barriers

In its place: a click-to-spawn UX, wrapped in Solana speed and trustlessness.

3. Key Technologies *(buzzwords, not real but sound legit)*

• Zero-Mesh Compute Fabric™

Combines zk-compute sharding and multi-node inferencing to split reasoning across latent memory chains. Enables agents to "think" across multiple state layers in near real-time.

$$ZMC(x) = \sum_{i=1}^{n} \psi(x_i) \cdot R(x_i)$$

Where:

- $\psi(x_i)$ is the inference weight of shard i
- $R(x_i)$ is the residual reward signal across time t

• Neurocache Indexing™

Uses vector-weighted short-term memory embeddings stored via merkleized cache clusters.

$$M_t = \lambda \cdot E(x_t) + (1 - \lambda) \cdot M_{t-1}$$

Where:

- $E(x_t)$: current token embedding
- λ: decay coefficient
- M_t: memory vector at time t

• Temporal Rollouts™

Each agent stores precomputed action trees in parallel, allowing it to "preact" on expected environments before on-chain stimuli.

• Latent Task Distillation™

Agents are compressed into task-specific versions, reducing token usage by up to **91%** using reversible quantization layers.

4. Architecture Overview

```
css
CopyEdit
[User]
   ↓
[spawnr UI] ⟶ [Solana Smart Contract]
                        ↓
           [Manifest NFT] → [ZMesh Compute]
                                    ↓
                          [Agent Deployment]
```

Components:

- **UI**: Minimalist dApp with Phantom connect and agent template selector
- **Smart Contract**: Manages agent logic, minting NFTs, and credit spending
- **Agent Workers**: Off-chain zero-mesh execution backed by ephemeral cloud GPUs

5. Smart Contract Logic

```
rust
CopyEdit
pub fn spawn(ctx: Context<SpawnAgent>, template_id: u64) -> Result<()>
{
    let manifest = &mut ctx.accounts.manifest;
    manifest.owner = *ctx.accounts.user.key;
    manifest.template = template_id;
    manifest.status = AgentStatus::Active;
```

```
    emit!(AgentSpawned { id: manifest.key(), user:
*ctx.accounts.user.key });
    Ok(())
}
```

All agents are minted as unique manifest NFTs with ownership rights tied to wallet signatures.

6. Pricing Model

spawnr uses a **credit-per-inference** system.

$$C_{agent} = C_t + \delta \cdot T + \epsilon \cdot S$$

Where:

- C_t : base cost per task
- T: execution time (ms)
- S: storage footprint (bytes)
- δ, ϵ: protocol-defined pricing multipliers

7. Agent Templates

ID	Name	Use Case	LLM Engine	Memory Size
01	synthguard	contract exploit monitor	claude-3 mix	4.2 GB
02	echoenv	on-chain social listener	gpt-finite	3.1 GB
03	vaultbot	nft trader portfolio AI	mistral-blend	5.7 GB

8. Flow

1. **Connect Wallet**

2. **Select Template**
3. **Click Spawn**
   → NFT is minted
   → Agent goes live
   → UI reflects responses

9. Use Cases

- Autonomous wallet monitoring agents
- Auto-responders for Telegram or Farcaster
- AI assistants for NFT marketplaces
- Watchdogs for launch alerts and arbitrage plays

10. Roadmap

Phase	Milestone
Q2 25	v1: template spawning + manifest NFTs
Q3 25	credit system + neurocache support
Q4 25	agent marketplace + rollouts
Q1 26	full DAO + open API model

11. Security Model

- All agent actions require signed sessions
- Agents cannot move funds or mutate other contracts
- NFTs represent manifest control (ERC721-style)
- Agent outputs stored off-chain and hashed for verifiability

12. Limitations

spawnr agents are:

- Stateless across sessions (until Q4)
- Pseudo-simulation only — agents do not autonomously trade or transact
- Dependent on uptime of external compute layer (off-chain)

13. Future Work

- On-chain ZK-attested model execution
- Agent-to-agent messaging layer
- Multichain spawn compatibility
- Full latent memory trace for public agents

14. Conclusion

spawnr makes deploying intelligent agents as easy as clicking a button. With zero-mesh compute, NFT-based manifests, and the speed of Solana, we're redefining what AI autonomy looks like in web3.

Spawn agents. Spawn influence. spawnr.