EXtream.AI Ready-to-run setup scripts for both **Windows (PowerShell .ps1)** and **macOS/Linux (.sh)**.

They:

- Create the standard **Extream.AI** directory tree
- Install core tooling (Python, Node, Yarn, Docker Desktop, kubectl, Minikube, Terraform, Helm)
- Create isolated dev environments (Python venv, Node deps)
- Provide one-command start/stop/clean helpers

You can copy each script into your repo's `/scripts` folder exactly as named below.

Windows (PowerShell) Scripts

Requirements: Windows 10/11. Run **PowerShell as Administrator** the first time (for tool installs).

`scripts\win\00_bootstrap.ps1`

```
# Creates the Extream.AI directory tree
$ErrorActionPreference = "Stop"

$root = Split-Path -Parent (Split-Path -Parent $MyInvocation.MyCommand.Path)
$dirs = @(
  "backend\app","backend\tests",
  "frontend\src",
  "ingestion\services","ingestion\configs",
  "agents\state_machines","agents\bots",
  "governance\opa","governance\rules","governance\dashboard",
  "eap\codecs","eap\adapters",
  "infra\docker","infra\k8s","infra\terraform",
  "monitoring\grafana","monitoring\prometheus",
  "data","docs","scripts\win","scripts\mac"
)

foreach ($d in $dirs) {
  $path = Join-Path $root $d
  if (-not (Test-Path $path)) { New-Item -ItemType Directory -Force -Path $path
| Out-Null }
}
Write-Host "✅ Directory structure created under $root"
```

`scripts\win\01_install_prereqs.ps1`

```
# Installs core tools via winget (preferred). Run as Administrator on first
install.
$ErrorActionPreference = "Stop"
```

```
function Ensure-Winget {
  if (-not (Get-Command winget -ErrorAction SilentlyContinue)) {
    Write-Error "winget not found. Install 'App Installer' from Microsoft Store
and re-run."
  }
}
Ensure-Winget

# Upgrade all
winget upgrade --all

# Git
winget install --id Git.Git -e --source winget

# Python (3.11)
winget install Python.Python.3.11

# Node LTS + Yarn
winget install OpenJS.NodeJS.LTS
npm install -g yarn

# Docker Desktop
winget install Docker.DockerDesktop

# Kubernetes CLIs
winget install Kubernetes.kubectl
winget install Kubernetes.minikube

# Terraform + Helm
winget install HashiCorp.Terraform
winget install Helm.Helm

Write-Host "✅ Core prerequisites installed. Log out/in if Docker Desktop was
newly installed."
```

scripts\win\02_setup_env.ps1

```
# Creates Python venv, installs backend deps; installs frontend deps.
$ErrorActionPreference = "Stop"
$Root = Split-Path -Parent (Split-Path -Parent $MyInvocation.MyCommand.Path)

# Backend (Python FastAPI)
$backend = Join-Path $Root "backend"
if (-not (Test-Path "$backend\requirements.txt")) {
  @"
fastapi
uvicorn[standard]
pydantic
python-dotenv
requests
"@ | Out-File -Encoding UTF8 "$backend\requirements.txt"
}
python -m venv "$backend\venv"
& "$backend\venv\Scripts\pip.exe" install --upgrade pip wheel setuptools
& "$backend\venv\Scripts\pip.exe" install -r "$backend\requirements.txt"
```

```
# Frontend (React)
$frontend = Join-Path $Root "frontend"
if (-not (Test-Path "$frontend\package.json")) {
  pushd $frontend
  npm init -y
  npm pkg set type="module"
  yarn add react react-dom
  yarn add -D vite
  (Get-Content package.json) -replace '"test": "echo.*',
'"dev":"vite","build":"vite build","preview":"vite preview"' | Set-Content
package.json
  popd
} else {
  pushd $frontend; yarn install; popd
}

Write-Host "✅ Environments ready (Python venv, frontend deps)."
```

scripts\win\03_start_dev.ps1

```
# Starts backend (uvicorn) and frontend (vite) in separate windows
$ErrorActionPreference = "Stop"
$Root = Split-Path -Parent (Split-Path -Parent $MyInvocation.MyCommand.Path)

# Ensure minimal FastAPI app exists
$backend = Join-Path $Root "backend"
if (-not (Test-Path "$backend\app\main.py")) {
  New-Item -ItemType Directory -Force -Path "$backend\app" | Out-Null
  @"
from fastapi import FastAPI
app = FastAPI()
@app.get("/health")
def health():
    return {"status":"ok"}
"@ | Out-File -Encoding UTF8 "$backend\app\main.py"
}

# Backend
$env:PYTHONPATH = "$backend"
Start-Process powershell -ArgumentList "-NoExit","-Command","&
`"$backend\venv\Scripts\python.exe`" -m uvicorn app.main:app --reload --port
8000"

# Frontend
$frontend = Join-Path $Root "frontend"
Start-Process powershell -ArgumentList "-NoExit","-Command","cd `"$frontend`";
yarn dev --port 3000"

Write-Host "✅ Dev services launching: Backend:8000, Frontend:3000"
```

scripts\win\04_stop_dev.ps1

```
# Stops common dev processes
$procs = "uvicorn","node","vite"
foreach ($p in $procs) {
  Get-Process $p -ErrorAction SilentlyContinue | Stop-Process -Force
-ErrorAction SilentlyContinue
}
Write-Host "🛑 Stopped common dev processes."
```

scripts\win\05_clean_env.ps1

```
# Removes node_modules and Python venv caches (safe clean)
$ErrorActionPreference = "Stop"
$Root = Split-Path -Parent (Split-Path -Parent $MyInvocation.MyCommand.Path)

# Backend
Remove-Item -Recurse -Force -ErrorAction SilentlyContinue "$Root\backend\venv"

# Frontend
Remove-Item -Recurse -Force -ErrorAction SilentlyContinue
"$Root\frontend\node_modules"

Write-Host "🧹 Clean complete."
```

How to run (Windows):

```
# Run PowerShell as Administrator (first time)
Set-ExecutionPolicy Bypass -Scope Process -Force
.\scripts\win\00_bootstrap.ps1
.\scripts\win\01_install_prereqs.ps1
.\scripts\win\02_setup_env.ps1
.\scripts\win\03_start_dev.ps1
```

Visit:

- Backend: http://localhost:8000/health
- Frontend: http://localhost:3000

---

macOS / 🐧 Linux (Shell) Scripts

Requirements: macOS 12+ (Intel/Apple Silicon) or Ubuntu 22.04+.
Make files executable once: `chmod +x scripts/mac/*.sh`

scripts/mac/00_bootstrap.sh

```
#!/usr/bin/env bash
set -euo pipefail
ROOT="$(cd "$(dirname "${BASH_SOURCE[0]}")"/../.. && pwd)"
mkdir -p "$ROOT/backend/app" "$ROOT/backend/tests" \
         "$ROOT/frontend/src" \
         "$ROOT/ingestion/services" "$ROOT/ingestion/configs" \
         "$ROOT/agents/state_machines" "$ROOT/agents/bots" \
         "$ROOT/governance/opa" "$ROOT/governance/rules"
"$ROOT/governance/dashboard" \
         "$ROOT/eap/codecs" "$ROOT/eap/adapters" \
         "$ROOT/infra/docker" "$ROOT/infra/k8s" "$ROOT/infra/terraform" \
         "$ROOT/monitoring/grafana" "$ROOT/monitoring/prometheus" \
         "$ROOT/data" "$ROOT/docs" "$ROOT/scripts/mac" "$ROOT/scripts/win"
echo "✅ Directory structure created under $ROOT"
```

scripts/mac/01_install_prereqs.sh

```
#!/usr/bin/env bash
set -euo pipefail

if [[ "$OSTYPE" == "darwin"* ]]; then
  # macOS via Homebrew
  if ! command -v brew >/dev/null 2>&1; then
    /bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
  fi
  brew update
  brew install git python@3.11 node kubectl minikube helm
  brew tap hashicorp/tap && brew install hashicorp/tap/terraform
  brew install --cask docker
else
  # Ubuntu/Debian
  sudo apt update
  sudo apt install -y git python3.11 python3.11-venv nodejs npm curl
apt-transport-https gnupg lsb-release
  # kubectl
  curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key | sudo gpg
--dearmor -o /etc/apt/trusted.gpg.d/kubernetes.gpg
  echo "deb https://pkgs.k8s.io/core:/stable:/v1.30/deb/ /" | sudo tee
/etc/apt/sources.list.d/kubernetes.list
  sudo apt update && sudo apt install -y kubectl
  # minikube
  curl -LO
https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
  sudo install minikube-linux-amd64 /usr/local/bin/minikube
  # terraform
  curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/hashicorp-archive-keyring.gpg
  echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee
/etc/apt/sources.list.d/hashicorp.list
  sudo apt update && sudo apt install -y terraform
  # helm
  curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 |
bash
fi

# Yarn
if ! command -v yarn >/dev/null 2>&1; then
  npm install -g yarn
fi
```

```
echo "✅ Core prerequisites installed. Start Docker Desktop (macOS) before
continuing."
```

scripts/mac/02_setup_env.sh

```
#!/usr/bin/env bash
set -euo pipefail
ROOT="$(cd "$(dirname "${BASH_SOURCE[0]}")"/../.. && pwd)"

# Backend (Python FastAPI)
cd "$ROOT/backend"
python3 -m venv venv
source venv/bin/activate
if [[ ! -f requirements.txt ]]; then
  cat > requirements.txt <<'EOF'
fastapi
uvicorn[standard]
pydantic
python-dotenv
requests
EOF
fi
pip install --upgrade pip wheel setuptools
pip install -r requirements.txt
deactivate

# Frontend (React)
cd "$ROOT/frontend"
if [[ ! -f package.json ]]; then
  npm init -y
  npm pkg set type="module"
  yarn add react react-dom
  yarn add -D vite
  node -e "let p=require('./package.json');p.scripts={dev:'vite',build:'vite
build',preview:'vite
preview'};require('fs').writeFileSync('package.json',JSON.stringify(p,null,2));
"
else
  yarn install
fi

echo "✅ Environments ready (Python venv, frontend deps)."
```

scripts/mac/03_start_dev.sh

```
#!/usr/bin/env bash
set -euo pipefail
ROOT="$(cd "$(dirname "${BASH_SOURCE[0]}")"/../.. && pwd)"

# Backend (ensure minimal app)
cd "$ROOT/backend"
source venv/bin/activate || true
if [[ ! -f app/main.py ]]; then
  mkdir -p app
  cat > app/main.py <<'PY'
```

```
from fastapi import FastAPI
app = FastAPI()
@app.get("/health")
def health(): return {"status":"ok"}
PY
fi
(uvicorn app.main:app --reload --port 8000 &) >/dev/null 2>&1

# Frontend
cd "$ROOT/frontend"
(yarn dev --port 3000 &) >/dev/null 2>&1

echo "✅ Dev services launching: Backend:8000, Frontend:3000"
```

scripts/mac/04_stop_dev.sh

```
#!/usr/bin/env bash
set -euo pipefail
pkill -f "uvicorn app.main" || true
pkill -f "vite" || true
echo "🛑 Stopped common dev processes."
```

scripts/mac/05_clean_env.sh

```
#!/usr/bin/env bash
set -euo pipefail
ROOT="$(cd "$(dirname "${BASH_SOURCE[0]}")"/../.. && pwd)"
rm -rf "$ROOT/backend/venv" "$ROOT/frontend/node_modules" 2>/dev/null || true
echo "🧹 Clean complete."
```

Quick Smoke Test

Backend

```
curl http://localhost:8000/health
# Expected → { "status": "ok" }
```

Frontend

Open `http://localhost:3000` → React splash screen.

Optional: `.env` templates

`backend/.env.example`

```
ENV=local
PORT=8000
API_BASE=/api
JWT_SECRET=change_me
```

frontend/.env.example

```
VITE_API_URL=http://localhost:8000
```

Drop these into `/scripts/win` and `/scripts/mac`, run `00_bootstrap` → `01_install_prereqs` → `02_setup_env` → `03_start_dev`, and you’re live.