GitHub Client Creation & Environment Variables Explained

This document explains how the GitHub client is created and how each environment variable in your `.env` file is used throughout the AI Reviewer project.

📋 Overview of Environment Variables

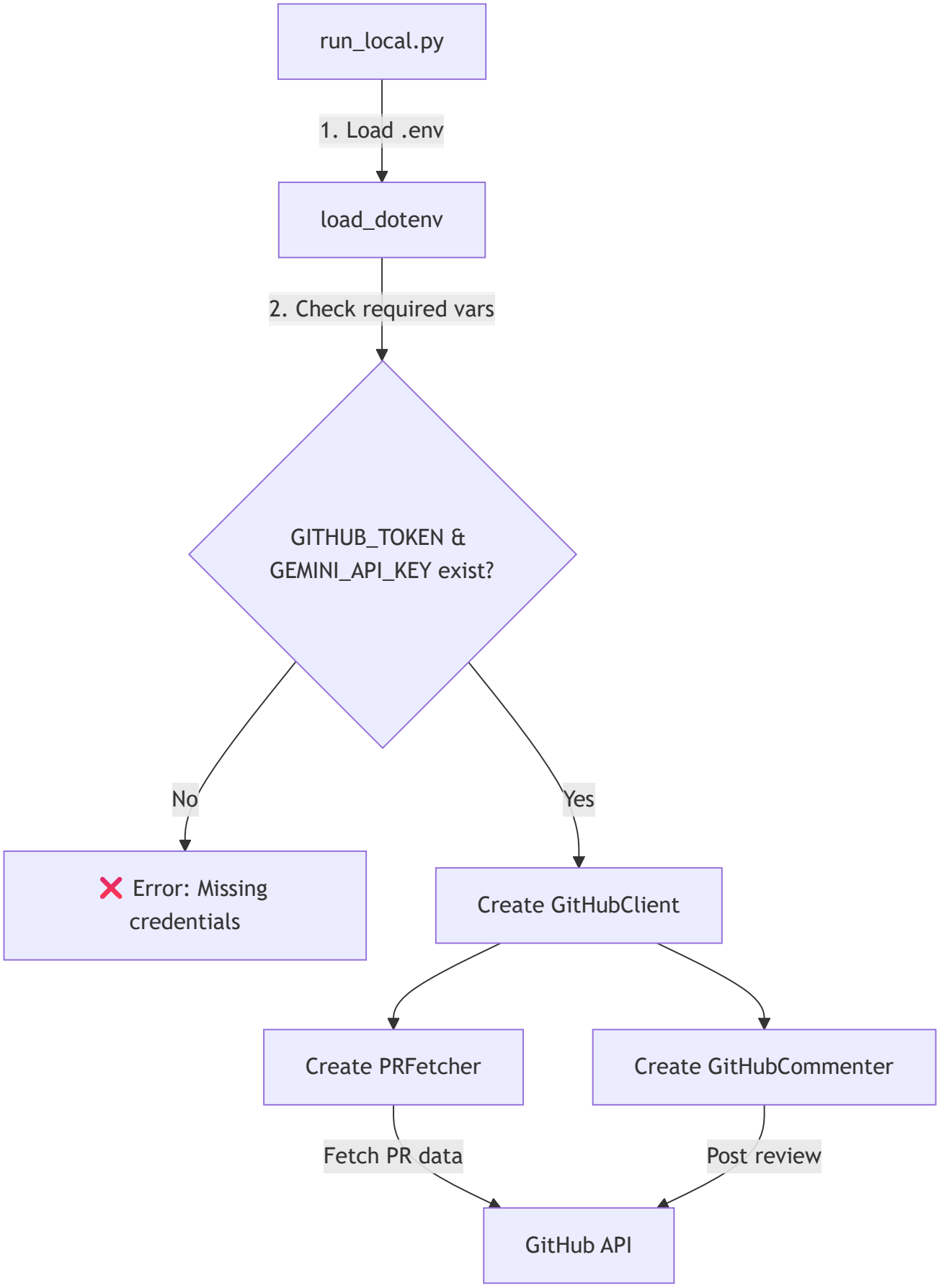
Your .env file contains several categories of variables:

Category	Variables	Purpose
AI/LLM	`GEMINI_API_KEY`, `GEMINI_MODEL`	Powers the AI code review agents
GitHub Auth (Token)	`GITHUB_TOKEN`	Personal Access Token for reading/writing to GitHub
GitHub Auth (App)	`GITHUB_APP_ID`, `GITHUB_PRIVATE_KEY_PATH`, `GITHUB_WEBHOOK_SECRET`, `INSTALLATION_ID`	GitHub App authentication (more secure, production-ready)
Database	`DATABASE_URL`	Stores review history
Server	`FASTAPI_HOST`, `FASTAPI_PORT`, `REDIS_URL`	Web server configuration
Configuration	`REVIEW_MODE`	Controls review behavior

🔑 GitHub Client Creation Flow

Entry Point: `run_local.py`

When you run `python run_local.py "VijayalakshmiKYADGI/test" 1`, here's what happens:



Code Location: run_local.py:24-26

```
if not os.getenv("GITHUB_TOKEN") or not os.getenv("GEMINI_API_KEY"):
    print("❌ ERROR: Missing GITHUB_TOKEN or GEMINI_API_KEY in .env file.")
    return
```

Code Location: run_local.py:31-33

```
 client = GitHubClient()
fetcher = PRFetcher(client)
commenter = GitHubCommenter(client)
```

🏗️ GitHubClient Architecture

The `GitHubClient` class supports **two authentication modes**:

Mode 1: Personal Access Token (PAT) - What you're using now

Code Location: github_integration/client.py:46-66

```
if auth_mode == "token":
    # Read GITHUB_TOKEN from environment
    self.token = token or os.getenv("GITHUB_TOKEN")

    if not self.token:
        raise ValueError("GitHub token not found...")

    # Initialize PyGithub client with token
    self.github = Github(self.token)

    # Create HTTP client for REST API calls
    self.http_client = httpx.AsyncClient(
        headers={
            "Authorization": f"token {self.token}",
            "Accept": "application/vnd.github.v3+json"
        }
    )
```

What happens:

1. Reads `GITHUB_TOKEN` from `.env` file
2. Creates a PyGithub client (for high-level operations like `get_repo()`, `get_pr()`)
3. Creates an async HTTP client (for low-level REST API calls like posting reviews)

Mode 2: GitHub App - **Advanced/Production mode**

Code Location: [github_integration/client.py:69-98]

```
elif auth_mode == "app":
    from .app_auth import GitHubAppAuth
    from config.app_config import GitHubAppConfig

    # Load GitHub App configuration
    config = GitHubAppConfig.from_env()  # Reads GITHUB_APP_ID, etc.

    # Initialize App authentication
    self.app_auth = GitHubAppAuth(
        app_id=config.app_id,
        private_key_path=config.private_key_path
    )
```

What happens:

1. Reads `GITHUB_APP_ID`, `GITHUB_PRIVATE_KEY_PATH`, `GITHUB_WEBHOOK_SECRET` from `.env`
2. Creates JWT tokens for GitHub App authentication
3. Gets installation tokens for specific repositories

> [!NOTE]
> Your project supports both modes, but defaults to **token mode** for simplicity. GitHub App mode is more secure and has higher rate limits, but requires more setup.

📍 Where Each Variable is Used

1. `GITHUB_TOKEN` (Personal Access Token)

Purpose: Authenticates API requests to GitHub for reading PRs and posting reviews.

Used in:

- github_integration/client.py:48 - Client initialization
- run_local.py:24 - Validation check

What it does:

- Reads public/private repository data
- Fetches PR diffs and metadata
- Posts review comments
- Approves/requests changes on PRs (if you have permissions)

Format: `ghp_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx` (classic token) or `github_pat_xxxxx` (fine-grained token)

Required Scopes:

- `repo` (full repository access)
- `pull_requests` (read/write PRs)

---

2. `GEMINI_API_KEY` (Google AI API Key)

Purpose: Powers the AI agents that analyze code and generate review comments.

Used in:

- core/crew.py:33 - CrewAI LLM initialization
- agents/security_agent.py:15 - Security analysis
- agents/code*quality*agent.py:15 - Code quality checks
- agents/performance_agent.py:15 - Performance analysis
- agents/architecture_agent.py:15 - Architecture review
- agents/comprehensive_agent.py:15 - Comprehensive review
- agents/report*aggregator*agent.py:13 - Report aggregation

Example usage in agents:

```
llm = ChatGoogleGenerativeAI(
    model="gemini-2.5-flash-lite",
    google_api_key=os.getenv("GEMINI_API_KEY"),
    temperature=0.1
)
```

---

3. `GEMINI_MODEL`

Purpose: Specifies which Gemini model to use for AI analysis.

Current value: `gemini-2.5-flash-lite`

Options:

- `gemini-2.5-flash-lite` - Fastest, cheapest (what you're using)
- `gemini-1.5-flash` - Balanced speed/quality
- `gemini-1.5-pro` - Highest quality, slower

---

4. `REVIEW_MODE`

Purpose: Controls whether the AI reviews only changed lines or entire files.

Current value: `full_file`

Used in: config/app_config.py:120

Options:

- `changes_only` - Review only the lines modified in the PR (faster)
- `full_file` - Review entire files that were touched (more thorough)

---

5. `GITHUB_APP_ID`, `GITHUB_PRIVATE_KEY_PATH`, `GITHUB_WEBHOOK_SECRET`

Purpose: GitHub App authentication (alternative to Personal Access Token).

Used in: config/app_config.py:41-55

When used:

- Only when `GitHubClient(auth_mode="app")` is explicitly set
- For production deployments with webhooks
- When you want higher rate limits (5000 requests/hour vs 1000 for PAT)

Current status: You have these configured, but the code defaults to `token` mode.

---

6. `INSTALLATION_ID`

Purpose: Links your GitHub App to specific repositories/organizations.

Current value: `104539256`

Used in: config/app_config.py:58-68

What it represents: The installation ID when you installed your GitHub App on your account/organization.

---

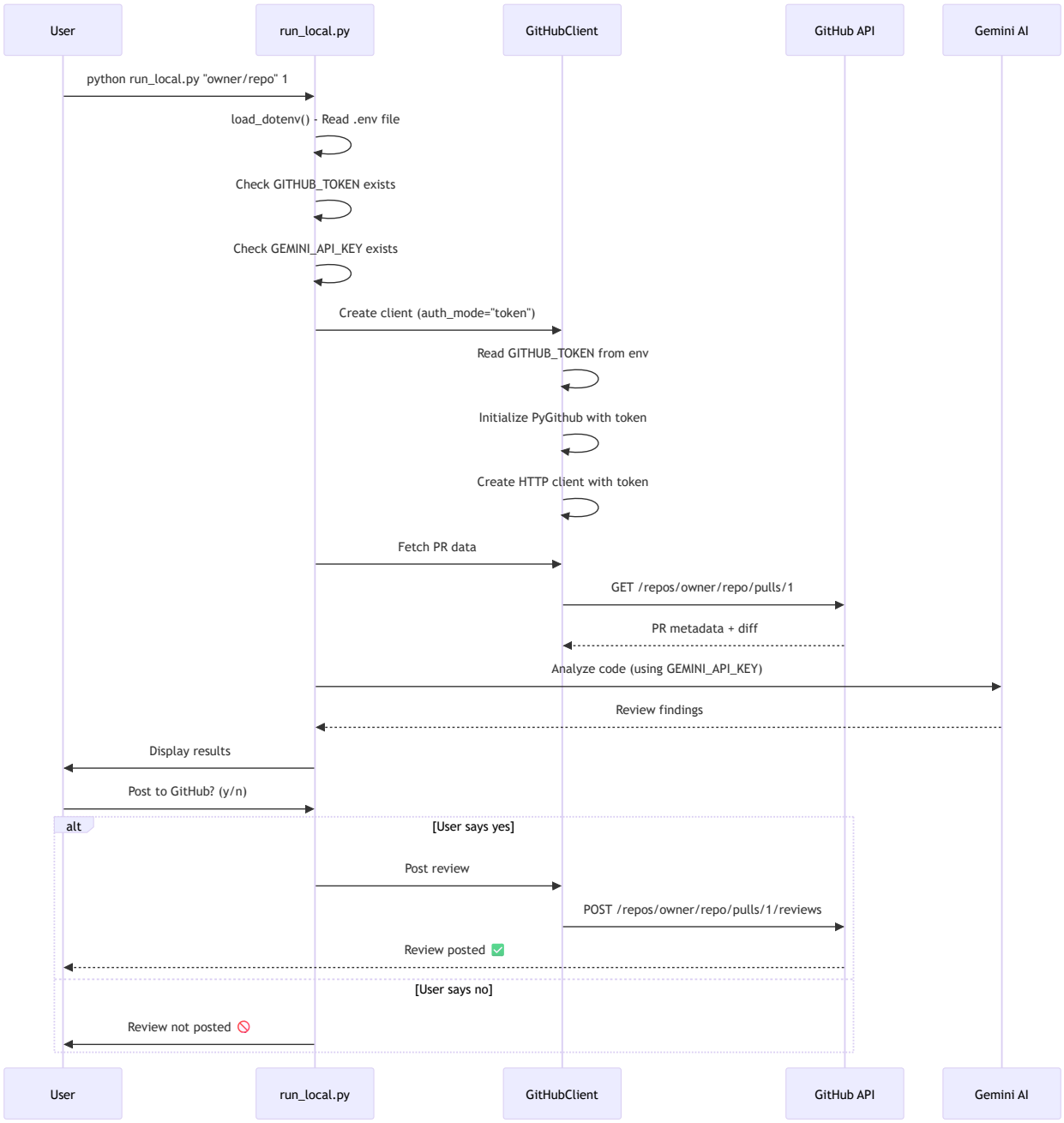
7. `DATABASE_URL`, `REDIS_URL`, `FASTAPI_HOST`, `FASTAPI_PORT`

Purpose: Backend infrastructure for the web server mode.

Used when: Running the FastAPI web server (`api/main.py`) instead of `run_local.py` .

Not used in local script mode - Only relevant for production deployments.

🔄 Complete Authentication Flow Diagram



🎯 Quick Reference: Variable Usage Summary

Variable	Required?	Used By	Purpose
`GITHUB_TOKEN`	✅ Yes	`GitHubClient`, `run_local.py`	Read/write GitHub PRs
`GEMINI_API_KEY`	✅ Yes	All AI agents, `core/crew.py`	Power AI analysis

Variable	Required?	Used By	Purpose
`GEMINI_MODEL`	⚠️ Optional	AI agents	Choose AI model (defaults to flash)
`REVIEW_MODE`	⚠️ Optional	Review pipeline	Control review scope
`GITHUB_APP_ID`	❌ No*	`GitHubAppAuth` (app mode only)	GitHub App authentication
`GITHUB_PRIVATE_KEY_PATH`	❌ No*	`GitHubAppAuth` (app mode only)	GitHub App private key
`GITHUB_WEBHOOK_SECRET`	❌ No*	Webhook handler	Verify webhook signatures
`INSTALLATION_ID`	❌ No*	`InstallationManager`	GitHub App installation
`DATABASE_URL`	❌ No**	FastAPI server	Store review history
`REDIS_URL`	❌ No**	Celery tasks	Background job queue
`FASTAPI_HOST`	❌ No**	FastAPI server	Web server host
`FASTAPI_PORT`	❌ No**	FastAPI server	Web server port

* Only required if using GitHub App mode (`auth_mode="app"`)
** Only required if running web server mode (not `run_local.py`)

💡 Key Takeaways

1. **For local script usage** (`run_local.py`), you only need:
   - `GITHUB_TOKEN` - To access GitHub
   - `GEMINI_API_KEY` - To run AI analysis
2. **GitHubClient is created** in run_local.py:31 and defaults to **token mode**.
3. **Token mode** reads `GITHUB_TOKEN` from `.env` and uses it for all GitHub API calls.

4. **App mode** is more advanced and uses `GITHUB_APP_ID` + private key for production deployments.

5. **Your current setup** uses token mode, which works perfectly for analyzing any public repository.