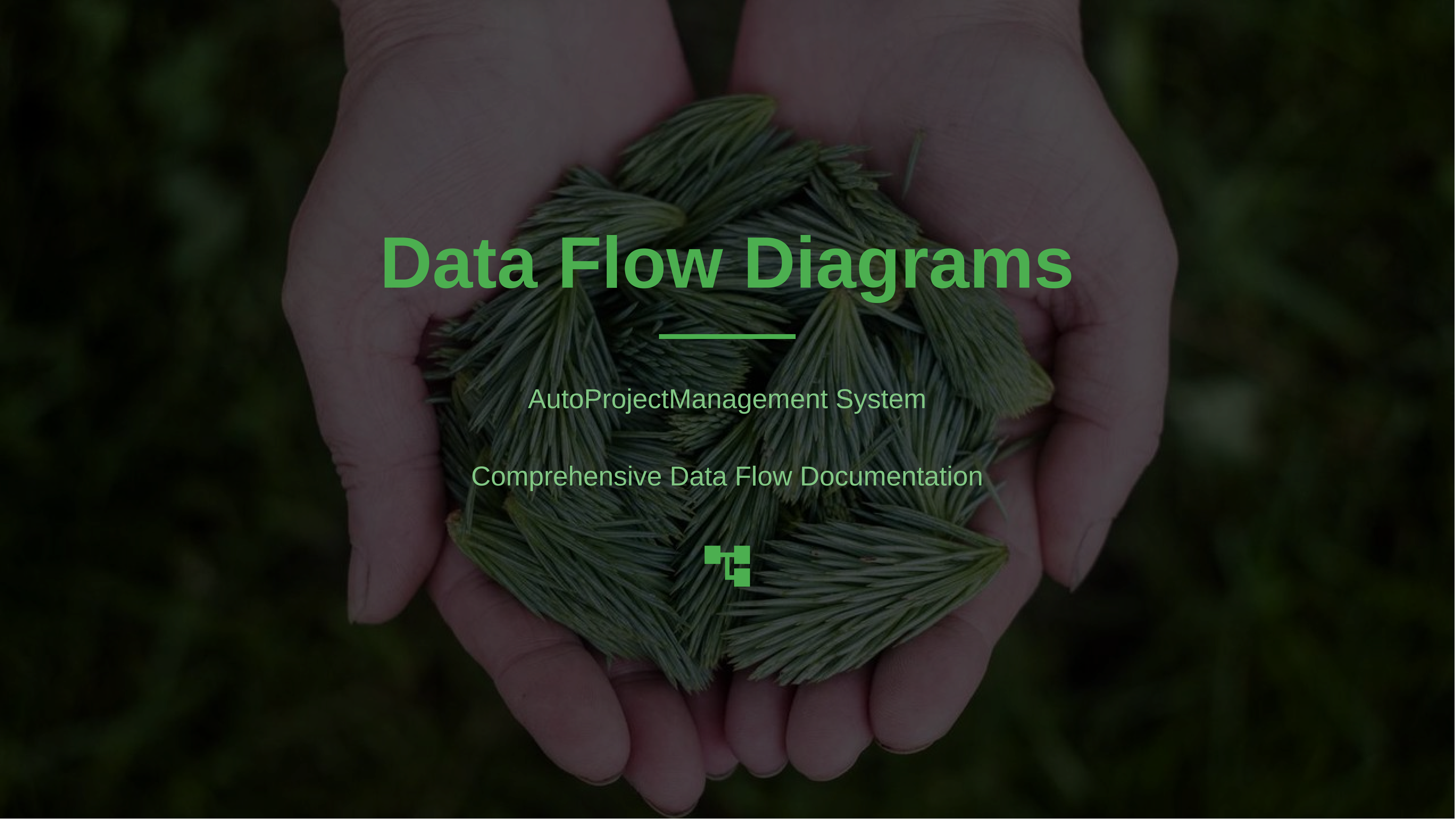


Data Flow Diagrams

AutoProjectManagement System

Comprehensive Data Flow Documentation

Overview

Document Purpose

Comprehensive Data Flow Diagrams for the AutoProjectManagement system, illustrating how data moves through various components and modules based on the actual implementation.

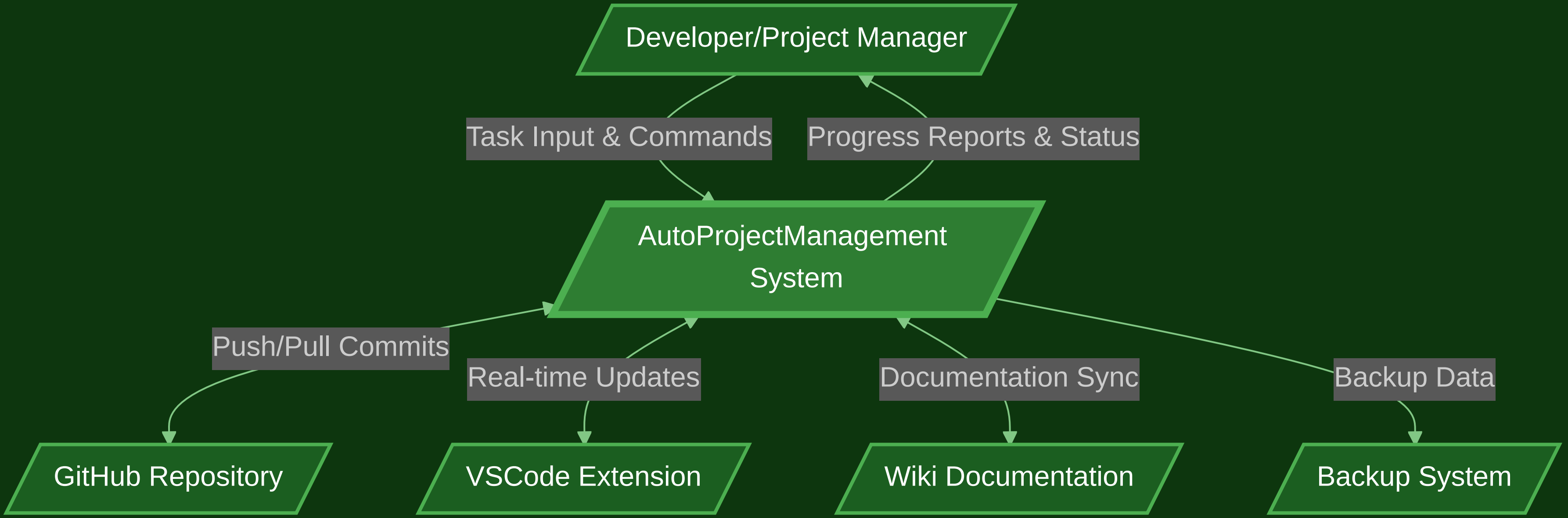
> This document provides a complete and accurate representation of the system's data flow, enabling better understanding and maintenance of the project management infrastructure.

Table of Contents

1. Context Diagram (Level 0)
2. Level 1 DFD - System Overview
3. Level 2 DFD - Core Modules
4. Level 3 DFD - Detailed Module Flows
5. Data Stores
6. Data Flow Descriptions

Context Diagram (Level 0)

High-level view of the AutoProjectManagement System and its interactions with external entities



⇄ Bidirectional Flows

System exchanges data with GitHub, VSCode, and Wiki in both directions

→ Unidirectional Flows

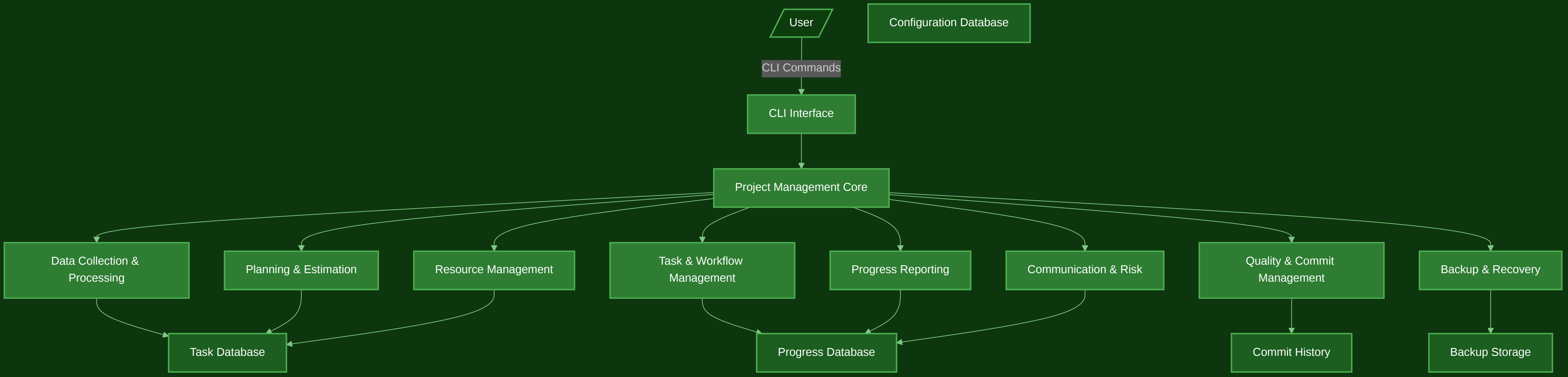
System receives input from Developer and sends data to Backup System

Central System

AutoProjectManagement acts as the central orchestrator of all data flows

Level 1 DFD - System Overview

Major processes and data stores of the AutoProjectManagement System



Central Orchestrator

P2 manages all project activities

Data Stores

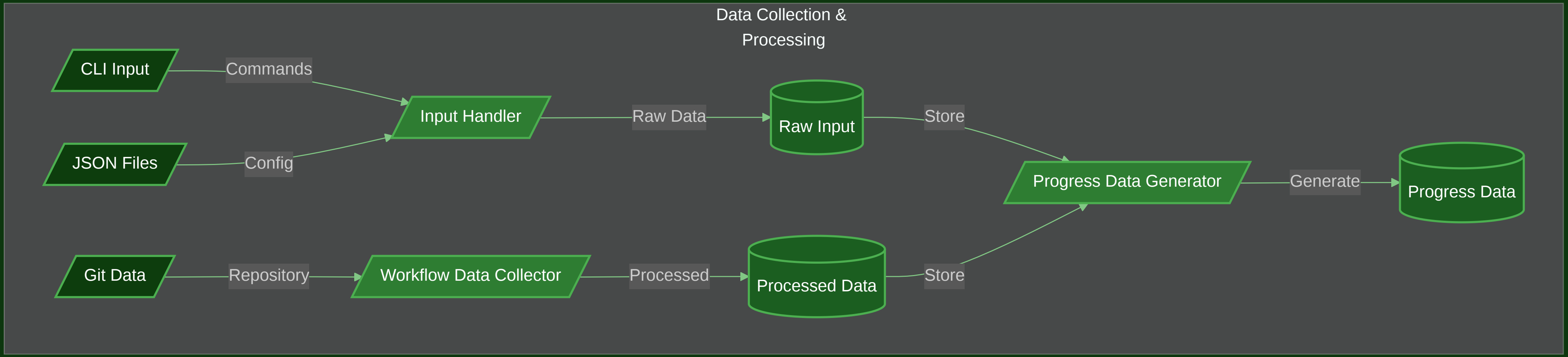
5 key databases for system operation

Process Flow

User → CLI → Project Core → Modules

Level 2 DFD - Data Collection & Processing

Detailed view of the Data Collection & Processing module, showing how input data is transformed into useful information



Input Sources

CLI commands, JSON files, and Git repository data

Processing Steps

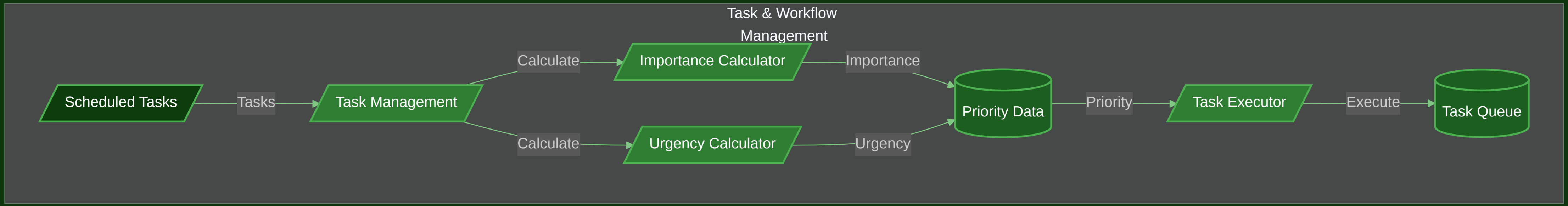
Input handling, workflow collection, and progress generation

Data Outputs

Raw input, processed data, and progress data stores

Level 2 DFD - Task & Workflow Management

Detailed view of the Task & Workflow Management module, showing how tasks are prioritized and executed



Task Management

Receives scheduled tasks and initiates priority calculation

Priority Calculation

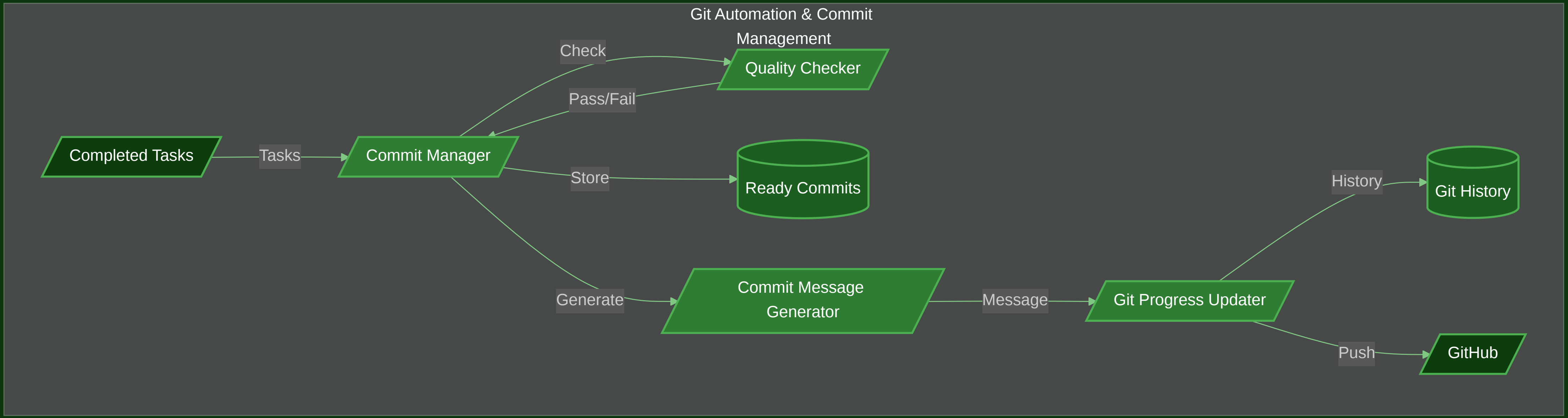
Evaluates importance and urgency to determine task priority

Task Execution

Executes tasks based on calculated priority and adds to queue

Level 2 DFD - Git Automation & Commit Management

Detailed view of the Git Automation & Commit Management module, showing how code commits are processed and managed



Quality Assurance

Code quality checking before commit approval

Message Generation

Automated commit message creation based on task details

Git Integration

Seamless push to GitHub with progress tracking

Data Stores

Primary data stores used by the AutoProjectManagement System

Task Database

Location:	JSonDataBase/Inputs/UserInputs/
Format:	JSON
Description:	User-defined tasks and configurations

```
{"task_id": "string", "task_name": "string", "description":
"string", "priority": "integer", "urgency": "integer",
"importance": "integer", "estimated_hours": "float",
"actual_hours": "float", "status": "string",
"dependencies": ["task_id"], "assigned_resources":
["resource_id"], "due_date": "date"}
```

Progress Database

Location:	JSonDataBase/OutPuts/
Format:	JSON
Description:	Calculated progress and status data

```
{"progress_id": "string", "task_id": "string",
"completion_percentage": "float", "hours_spent": "float",
"status": "string", "last_updated": "datetime"}
```

Configuration Database

Location:	autoproject_configuration.py
Format:	Python
Description:	System configuration parameters

Commit History

Location:	.git/ directory
Format:	Git
Description:	Git commit history and metadata

Backup Storage

Location: backups/ | Format: ZIP/JSON | System backups and archives

Data Flow

Information flows between stores through system processes

Data Integrity

Consistent schemas ensure reliable data handling

Data Flow Descriptions and Implementation Mapping

Primary data flows and their implementation mapping in the AutoProjectManagement System

⇄ Primary Data Flows

Flow ID	Flow Name	Frequency
F1	Task Input	On-demand
F2	Validated Tasks	Real-time
F3	WBS Structure	On task creation
F4	Progress Update	Continuous
F5	Commit Data	On task completion
F6	Backup Request	Scheduled
F7	Resource Allocation	On task scheduling

<> Implementation Mapping

CLI Interface
`autoprojectmanagement/cli.py`

Project Management Core
`autoprojectmanagement/main_modules/project_management_system.py`

Data Collection
`autoprojectmanagement/main_modules/data_collection_processing/`

Task Management
`autoprojectmanagement/main_modules/task_workflow_management/`

Backup System
`autoprojectmanagement/services/automation_services/backup_manager.py`

ⓘ Each data flow connects specific system components, enabling seamless information exchange throughout the project management lifecycle