Token Counter and API Key Management Features

Overview

Added comprehensive token usage tracking, cost estimation, and API key management to the PSScript AI service.

Features

1. Token Counter & Price Estimator

Real-Time Token Tracking

- Tracks all AI API calls automatically
- Records input tokens, output tokens, and total usage
- Calculates costs based on model pricing

Pricing Support (January 2026)

Supports all current OpenAI models with accurate pricing:

Model	Input (per 1M)	Output (per 1M)
GPT-5.2-Codex	$35	$70
GPT-5.2	$30	$60
GPT-5.2-Instant	$15	$30
GPT-4o	$2.50	$10
text-embedding-3-large	$0.13	-
text-embedding-3-small	$0.02	-

Usage Analytics

- Total tokens used across all sessions
- Total cost in USD
- Breakdown by model
- Recent session history (last 1000 sessions)
- Persistent storage (survives restarts)

2. API Key Management

Interactive Prompts

- Automatically prompts for API key if not found
- User-friendly setup wizard with instructions
- Validates key format (sk-prefix)
- Saves to `.env` file automatically

Secure Storage

- Keys saved to `.env` file in project root
- File permissions set to 600 (owner read/write only)
- Never logged or displayed in full

Key Operations

- Set new key
- Update existing key
- Test key validity
- Check key status
- Remove key

API Endpoints

Token Usage Endpoints

GET `/api/token-usage/summary`

Get summary of all token usage and costs.

Response:

```
{
  "total_tokens": 15230,
  "total_cost": 1.25,
  "formatted_cost": "$1.25",
  "by_model": {
    "gpt-5.2-codex": {
      "total_tokens": 15230,
      "total_cost": 1.25,
      "input_tokens": 10230,
      "output_tokens": 5000,
      "calls": 42
    }
  },
  "last_updated": "2026-01-09T09:51:53.946022",
  "session_count": 42
}
```

GET `/api/token-usage/recent?limit=10`

Get recent token usage sessions.

Query Parameters: - `limit` (optional): Number of sessions to return (1-100, default 10)

Response:

```
{
  "sessions": [
    {
      "timestamp": "2026-01-09T09:45:32.123456",
      "model": "gpt-5.2-codex",
      "operation": "PowerShell analysis",
      "input_tokens": 1000,
      "output_tokens": 500,
      "total_tokens": 1500,
      "cost": 0.0395
    }
  ]
}
```

POST `/api/token-usage/estimate`

Estimate cost before making an API call.

Request:

```
{
  "model": "gpt-5.2-codex",
  "input_text": "Your PowerShell script or prompt text",
  "estimated_output_tokens": 500
}
```

Response:

```
{
  "model": "gpt-5.2-codex",
  "estimated_input_tokens": 13,
  "estimated_output_tokens": 500,
  "estimated_total_tokens": 513,
  "estimated_cost": 0.03039,
  "formatted_cost": "$0.0304"
}
```

POST `/api/token-usage/reset`

Reset all token usage data (use with caution).

Response:

```
{
  "message": "Token usage data reset successfully"
}
```

API Key Management Endpoints

GET `/api/key/status`

Check if API key is configured.

Response:

```
{
  "configured": true,
  "masked_key": "sk-proj...zqAA",
  "mock_mode": false
}
```

POST `/api/key/set`

Set or update the OpenAI API key.

Request:

```
{
  "api_key": "sk-proj-your-api-key-here"
}
```

Response:

```
{
  "message": "API key saved successfully",
  "masked_key": "sk-proj...here"
}
```

POST `/api/key/test`

Test if the current API key is valid.

Response:

```
{
  "valid": true,
  "message": "API key is valid"
}
```

Command-Line Usage

API Key Management

```
 # Set API key interactively
python src/ai/utils/api_key_manager.py set

# Test API key
python src/ai/utils/api_key_manager.py test

# Update API key
python src/ai/utils/api_key_manager.py update

# Remove API key
python src/ai/utils/api_key_manager.py remove
```

Token Counter

```python
from utils.token_counter import token_counter

# Track usage
tokens, cost = token_counter.track_usage(
    model="gpt-5.2-codex",
    input_tokens=1000,
    output_tokens=500,
    operation="PowerShell analysis"
)

# Estimate cost
estimate = token_counter.estimate_cost(
    model="gpt-5.2-codex",
    estimated_input_tokens=2000,
    estimated_output_tokens=1000
)

# Get summary
summary = token_counter.get_usage_summary()
```

Files Created

1. `src/ai/utils/token_counter.py`
2. TokenCounter class
3. Cost calculation logic

4. Usage tracking and analytics

5. `src/ai/utils/api_key_manager.py`

6. APIKeyManager class
7. Interactive prompts

8. .env file management

9. `src/ai/utils/__init__.py`

10. Package initialization
11. Convenience exports

Integration

The features are automatically integrated into the AI service:

1. **Startup**:
2. API key manager checks for key on startup
3. Prompts interactively if missing (terminal mode)
4. Falls back to mock mode if no key provided
5. **Token Tracking**:
6. All AI API calls are automatically tracked
7. Usage data persisted to `token_usage.json`
8. No manual intervention required
9. **Web Interface**:
10. API endpoints available for frontend integration
11. Real-time usage monitoring
12. Cost estimation before expensive operations

Benefits

Cost Control

- Know exactly how much each operation costs
- Estimate before running expensive queries
- Track spending over time
- Set budgets and alerts (future feature)

Easy Setup

- No manual .env editing required
- Interactive setup wizard
- Clear instructions for obtaining API keys
- Validation to prevent mistakes

Visibility

- See which models cost the most
- Identify expensive operations
- Optimize based on actual usage
- Historical tracking for analysis

Future Enhancements

1. **Budget Alerts**
2. Set spending limits
3. Email/webhook notifications

4. Automatic throttling

5. **Advanced Analytics**

6. Cost per user/session
7. Time-based analysis
8. Model comparison charts

9. Optimization recommendations

10. **Frontend Integration**

11. Usage dashboard in web UI
12. Real-time cost display
13. Historical graphs

14. Budget management UI

15. **Rate Limiting**

16. Prevent runaway costs
17. Per-user/per-session limits
18. Graceful degradation

Created: January 9, 2026 **Status**: ✅ Deployed and Tested **Version**: 1.0.0

Generated 2026-01-16 23:34 UTC