Google Ads MCP - Design Documentation

Overview

The Google Ads Message Control Protocol (MCP) is a middleware solution designed to integrate Google Ads functionality with n8n workflow automation platform. This tool enables automated campaign management, performance monitoring, and optimization through a RESTful API interface.

System Architecture

High-Level Architecture

```
[n8n Platform] <-> [Google Ads MCP] <-> [Google Ads API]
```

The system consists of three main components: 1. **API Layer**: RESTful endpoints for interaction with n8n 2. **Service Layer**: Business logic and Google Ads API integration 3. **Monitoring Layer**: Logging and performance monitoring

Components

1. API Layer

- Express.js REST API
- Endpoint groups:
  - Bid and Budget Control
  - Campaign Performance
  - Creative Performance
- Input validation and error handling

2. Service Layer

- Google Ads API integration
- Campaign management logic
- Performance analysis
- Data transformation

3. Monitoring Layer

- Winston logger implementation
- Performance metrics tracking
- Error monitoring and reporting

API Endpoints

Bid and Budget Control

```
POST /api/v1/bid-budget/update
```

```
GET /api/v1/bid-budget/status
```

Campaign Performance

```
GET /api/v1/campaign/performance
GET /api/v1/campaign/metrics
```

Creative Performance

```
GET /api/v1/creative/performance
GET /api/v1/creative/metrics
```

Data Flow

1. **Request Flow**

   ```
   n8n -> MCP API -> Service Layer -> Google Ads API
   ```

2. **Response Flow**

   ```
   Google Ads API -> Service Layer -> Data Transform -> MCP API -> n8n
   ```

Security

1. **Authentication**
   - Environment-based configuration
   - Secure credential storage
   - Token management
2. **Data Protection**
   - HTTPS encryption
   - Input sanitization
   - Error message sanitization

Error Handling

1. **API Layer**
   - Input validation
   - Request format verification
   - Response status codes
2. **Service Layer**
   - Google Ads API error handling
   - Retry mechanisms
   - Error logging

Performance Considerations

1. **Optimization**
   - Connection pooling

- Response caching
- Rate limiting

2. **Monitoring**
   - Request/response timing
   - Error rate tracking
   - Resource usage monitoring

Integration Examples

n8n Workflow Example

```
// Example n8n workflow
{
  "nodes": [
    {
      "parameters": {
        "resource": "campaign",
        "operation": "getPerformance",
        "campaignId": "123456789"
      },
      "name": "Google Ads MCP",
      "type": "n8n-nodes-base.googleAdsMcp",
      "typeVersion": 1,
      "position": [
        880,
        300
      ]
    }
  ]
}
```

Deployment

1. **Requirements**
   - Node.js v14+
   - PM2 process manager
   - Environment configuration
2. **Setup Process**
   - Environment configuration
   - Dependency installation
   - Service initialization

Maintenance and Support

1. **Monitoring**
   - Performance metrics

   - Error tracking
   - Usage statistics
2. **Updates**
   - Version control
   - Changelog maintenance
   - Update procedure

Future Enhancements

1. **Planned Features**
   - Advanced bidding strategies
   - Machine learning optimization
   - Extended reporting capabilities
2. **Scalability**
   - Load balancing
   - Horizontal scaling
   - Performance optimization