Cross-Platform Social Media Post Automation Using Model Context Protocol (MCP): A Comprehensive Review and Proposed Architecture

Siddhant Lilhare[1], Pankaj Ubale[2], Gaurav Kanoje[3], Prof. Swapnil Warade[4]
[1,2,3]*Department of Information Technology, JD College of Engineering and Management, Dr. Babasaheb Ambedkar Technological University Nagpur, Maharashtra, India*
[4]*Guide, JD College of Engineering and Management, Dr. Babasaheb Ambedkar Technological University Nagpur, Maharashtra, India*

***Abstract*—Social media platforms serve as primary infrastructure for personal branding, enterprise communication, and community engagement, with active user identities exceeding 5.66 billion globally as of late 2025 [1], [2]. Despite this reach, producing platform-appropriate content for X (Twitter), Instagram, LinkedIn, and Facebook simultaneously remains highly labourintensive due to divergent tone expectations, character constraints, media requirements, and API complexities. Current automation solutions address only parts of this problem and generally lack AI-driven content intelligence, standardised tool integration, or meaningful privacy safeguards.**

This paper reviews the literature underpinning a novel pro-posed system: *AutoPost-MCP*, an AI-driven, multi-agent social media automation framework built around the Model Context Protocol (MCP). Users supply a natural-language content intent; specialised agents then generate platform-tailored post variants and publish them securely via a unified interface authenticated through Google OAuth 2.0. The survey spans social media automation history, LLM-based content generation, multi-agent system design, MCP's architecture and 2025 ecosystem growth, platform APIs with current rate limits, privacy mechanisms, and gaps in existing tools. Design choices are justified with reference to the surveyed literature, and research gaps, ethical considerations, and future directions are discussed.

***Index Terms*—Model Context Protocol, social media Automation, AI Agents, Cross-Platform Publishing, Google OAuth 2.0, Large Language Models, Multi-Agent Systems, Platform APIs**

I. INTRODUCTION

By October 2025, active social media identities had reached approximately 5.66 billion, corresponding to roughly 68.7% of the world's population and reflecting year-on-year growth of close to 5% [1], [2]. For content creators, digital marketers, entrepreneurs, and enterprises, this scale demands simultaneous engagement across channels that each carry distinct communicative cultures. X favours concise, timely commentary; Instagram rewards visually-driven storytelling with descriptive captions; LinkedIn values professional insights and career-relevant framing; Facebook prioritises community discussion and shareable links. Writing and publishing manually tailored versions of each message for every platform is neither scalable nor efficient.

Advances in large language models (LLMs) have made it feasible to delegate content drafting to an AI system that understands intent and can adapt expression to a target context [6]. Pairing this capability with agentic reasoning frameworks—where an AI model alternates between thinking and acting autonomously [20]—creates a pathway toward fully automated multi-platform publishing. The remaining challenge has been the absence of a standardised mechanism for agents to interact reliably and securely with the diverse external APIs they need to reach.

The Model Context Protocol (MCP), open-sourced by Anthropic in November 2024, addresses precisely this gap by defining a universal, vendor-neutral interface through which any MCP-compatible AI host can discover and invoke tools exposed by MCP-

compatible servers [3]. By late 2025 the protocol had been adopted across numerous LLM providers, developer tools, and cloud platforms, and its governance was transferred to the Agentic AI Foundation under the Linux Foundation, signalling its maturation as an open community standard [4], [5].

This paper synthesises these threads—LLM content generation, multi-agent architectures, MCP, platform APIs, and secure authentication—to motivate and detail the proposed AutoPost-MCP system.

A. Review Objectives

This paper aims to:

- Survey the historical progression and current landscape of social media automation.
- Examine MCP's architecture, security model, and 2025 adoption in detail.

AutoPost-MCP System Architecture

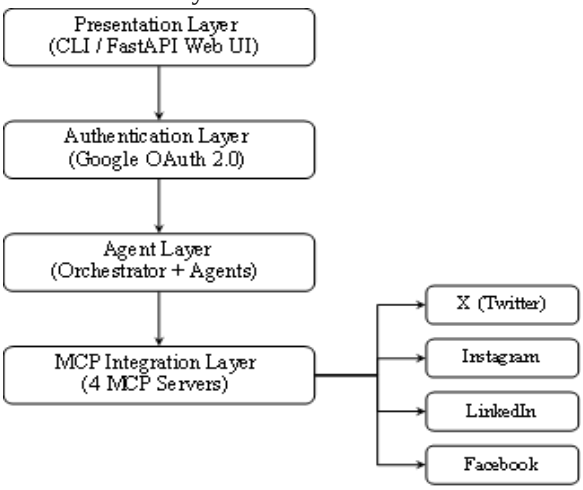

Fig. 1: Layered Architecture of the proposed AutoPost-MCP system.

- Analyse multi-agent patterns suited to content generation workflows.
- Review major platform APIs including updated rate limits for 2025–2026.
- Evaluate privacy and authentication best practices.
- Identify gaps in existing commercial and open-source tools.
- Justify the design rationale of AutoPost-MCP and map future research directions.

B. Paper Organisation

Section II covers background and related work. Section III surveys MCP technically. Section IV explores multi-agent architectures. Section V analyses platform APIs. Section VI addresses privacy. Section VII evaluates existing tools. Section VIII details the proposed system. Section IX covers challenges and ethics. Section X outlines gaps and future work. Section XI concludes.

II. BACKGROUND AND RELATED WORK

A. Evolution of Social Media Automation

Automation support for social media has developed across three recognisable phases. In the first phase, tools such as Hootsuite and Buffer offered calendar-based scheduling of manually composed posts [29]. These platforms reduced repetitive clicking but placed the full creative burden on the user; no AI assistance was available for drafting content.

Second-phase systems—Zapier and IFTTT being canonical examples—introduced conditional, trigger-based workflows that could, for instance, automatically cross-post a new blog article to Twitter when it was published [30]. While more dynamic, these systems remained rule-driven and could not adapt content to a new platform's conventions; a tweet and a LinkedIn post would carry identical text.

The third and current phase integrates generative AI directly into the posting pipeline. However, most implementations in this space are tightly coupled to a single LLM vendor or platform, use ad hoc function-calling patterns for tool integration, and offer limited attention to privacy. MCP creates a structural opportunity to move beyond this fragmentation toward a reusable, interoperable agent-tool ecosystem.

B. AI-Generated Content for social media

Transformer-based LLMs have demonstrated strong capability in producing contextually appropriate short-form text when guided by platform context, audience profiles, and stylistic constraints [6]. A recurring finding is that generic, undifferentiated AI output tends to be perceived as inauthentic: it lacks the idiomatic patterns, tonal registers, and structural rhythms that characterise organic posts on each platform [11], [12]. Research consistently indicates that per-platform prompt specialisation—encoding character limits, preferred emoji us-age, hashtag density, and opening-line conventions—yields more naturalistic outputs with higher perceived credibility.

This insight directly motivates the per-platform

content adaptation strategy in AutoPost-MCP: rather than generating a single post and distributing it verbatim, the Content Agent is prompted separately for each target platform.

C. Multi-Agent Systems in Content Workflows

Decomposing complex tasks among specialised agents is a well-established principle in both classical multi-agent re-search and contemporary LLM frameworks [15]– [17]. For content pipelines, a common decomposition assigns an orchestrator agent to interpret user goals and coordinate workflow, a content-generation agent to produce drafts, and a publisher agent to handle API interactions. This separation of concerns improves error localisation, allows parallel processing where appropriate, and makes individual components easier to test and replace.

The ReAct reasoning paradigm [20]—in which an agent alternates between *reasoning* traces (deciding what to do next) and *acting* steps (calling a tool and observing the result)—has become a standard pattern for LLM agents that need to interact with external services. Its explicit interleaving of thought and action substantially improves reliability over approaches that attempt either pure chain-of-thought or direct tool invocation without intermediate reasoning.

LangChain and its graph-oriented extension LangGraph pro-vide Python abstractions for building such agent pipelines, including memory management, cyclic workflow support, and—since 2025—first-class MCP tool registration [23].

D. Privacy Considerations

The Cambridge Analytica incident illustrated the systemic risk of allowing broad, unscoped data access to third-party applications operating on social platforms [25]. In its aftermath, the industry accelerated adoption of OAuth 2.0 as the canonical framework for delegated, scoped authorisation, ensuring that a third-party tool can be granted exactly the permissions it needs—and no more—without ever handling the user's raw credentials [26]. Google OAuth 2.0 extends this with OpenID Connect (OIDC) for identity verification, providing a trusted, well-audited authentication entry point that reduces friction for end users.

III. MODEL CONTEXT PROTOCOL: TECHNICAL SURVEY

A. Overview

MCP was open-sourced by Anthropic in November 2024 with the goal of standardising how AI agents connect to external data sources and tools [3]. Where traditional LLM function-calling required developers to hand-code integration logic specific to each model provider, MCP externalises this logic into independently deployable server processes that any compliant host can discover and invoke. By late 2025 a major specification update had been released, governance had been transferred to the Linux Foundation-affiliated Agentic AI Foundation, and thousands of community-contributed MCP servers existed for services ranging from databases to productivity software [4], [5].

B. Architecture

MCP defines three roles. The Host is the application containing the AI model (in AutoPost-MCP, the Python backend). The Client is a protocol-layer component embedded in the host that manages one or more server connections. Each Server is a separate process that exposes a schema-described catalogue of *tools* (callable functions), *resources* (readable data), and *prompts* (reusable instruction templates). Transport options include local standard I/O and remote HTTP with Server-Sent Events (SSE), giving deployers flexibility between tightly coupled local services and loosely coupled remote microservices.

TABLE I: Traditional Function Calling vs. MCP (2025–2026)

Aspect	Trad. Function Calling	MCP
Standardisation	Provider-specific schema	Open community standard
Interoperability	Single-vendor lock-in	Cross-LLM compatible
Tool Discovery	Static, compile-time	Dynamic server advertisement
Security	Developer-managed	Built-in sandboxing & consent
Deployment	In-process	Out-of-process microservices
Ecosystem (2026)	Fragmented, ad hoc	Thousands of servers; Linux Foundation governance

C. Security Model

MCP enforces several security principles at the protocol level. *Least privilege* restricts each server to advertising only the tools it explicitly declares, preventing implicit scope creep. *Process isolation* limits the blast radius of a compromised server component. *Explicit consent* requires configurable approval for tool invocations, enabling human-in-the-loop check-points. *Auditability* is built into the protocol exchange, producing logs of every tool call and its result without additional instrumentation.

D. Implementation in AutoPost-MCP

For the proposed system, four MCP servers are built—one per social platform—each wrapping the platform's SDK in a set of high-level tools (e.g., post_tweet, publish_instagram_container). The servers handle OAuth token refresh, rate-limit back-off, and error recovery internally, shielding the agent layer from platform-specific concerns.

IV. MULTI-AGENT AI ARCHITECTURE

A. Agent Roles

AutoPost-MCP uses three core agents operating in a directed graph managed by LangGraph:

- Orchestrator Agent:

Receives the user's natural-language intent, determines which platforms are targeted, and coordinates the downstream agents.

- Content Agent:

Executes a ReAct reasoning loop, querying the LLM with per-platform system prompts to generate four adapted post variants simultaneously.

- Publisher Agent:

After user approval, calls the appropriate MCP servers, handles retry on transient failures, and aggregates success/failure results.

B. Memory Management

Three memory layers support the agent pipeline. *Working memory* is the active LLM context window, carrying the cur-rent task state and recent tool results. *Episodic memory* persists a log of previous posts, enabling the system to avoid repetitive content and track publishing history. *Semantic memory* stores user-defined preferences such as brand voice guidelines, preferred hashtag sets, and audience profiles, making successive sessions progressively more personalised [15].

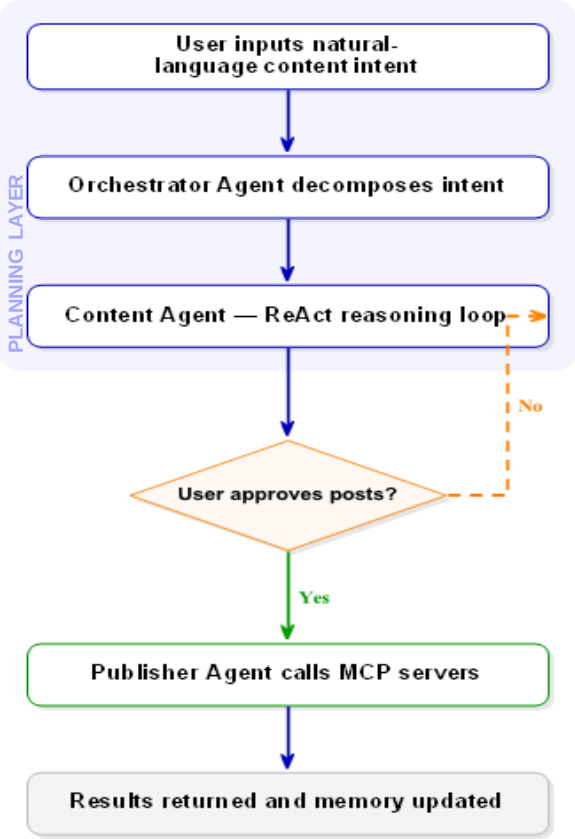

Fig. 2: Multi-agent workflow with user-in-the-loop approval.

C. Platform-Specific Content Adaptation

Rather than generating a single post and distributing it unchanged, the Content Agent is invoked with a dedicated system prompt for each platform. These prompts encode platform-native conventions:

- X: Under 280 characters, conversational tone, 1–2 topical hashtags, optional call-to-action.
- Instagram: Engaging opening hook, narrative voice, emoji usage, 5–10 hashtags appended after the caption.
- LinkedIn: Professional register, 150–300 words, insight-driven opening sentence, minimal hashtags.
- Facebook: Community-oriented tone, moderate length, explicit invitation to comment or share.

Fig. 3 illustrates how a single user intent produces four distinct post variants.

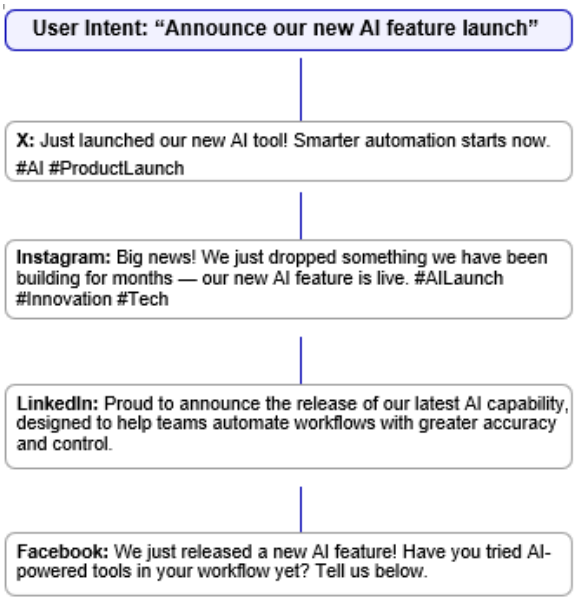

Fig. 3: Single intent adapted into four platform-native post variants.

V. SOCIAL MEDIA PLATFORM APIS

A. X (Twitter) API v2

The X API v2 supports programmatic tweet creation, media upload, and analytics retrieval [31]. Authentication requires OAuth 2.0 with PKCE for user-context operations. Free and Basic tier access imposes strict monthly posting quotas; Pro and Enterprise tiers or pay-per-use billing unlock higher limits. The AutoPost-MCP server for X wraps the POST/2/tweets and media-upload endpoints via Tweepy.

B. Instagram Graph API

Instagram's Graph API requires a linked Facebook Business Page and a User Access Token scoped to instagram_content_publish [32]. Content publication follows a two-step process: create a media container (image/video URL plus caption), then publish the container. The platform imposes an approximate ceiling of 100 container publications per rolling 24-hour window. Because Instagram does not support caption-only posts, the AutoPost-MCP con-tent agent always associates a visual description hint with the Instagram variant for future media attachment.

C. LinkedIn API

LinkedIn's ugcPosts endpoint supports text, image, and video content for personal profiles and Company Pages [33]. Required OAuth 2.0 scopes include w_member_social. The API enforces daily call ceilings that reset at midnight UTC. LinkedIn's content review algorithms apply strict spam-detection heuristics, making platform-authentic post style—long-form professional content rather than terse promotional text—essential.

D. Facebook Graph API

The Facebook Graph API allows posting text, links, and media to Pages and Groups via /{page-id}/feed [34]. A Page Access Token is obtained by exchanging a User Access Token holding the pages_manage_posts permission. Apps seeking extended permissions for public Page posting must pass a Meta App Review process, which should be budgeted as several working days during development.

TABLE II: Social Media Platform API Comparison (2025–2026)

Feature	X	Instagram	LinkedIn	Facebook
Max Text	280	2,200	3,000	63,000+
Media	Optional	Required	Optional	Optional
Post Limits	Tier-based	~100/24h	Daily reset	Tier-based
Auth	OAuth 2.0+PKCE	OAuth 2.0	OAuth 2.0	OAuth 2.0
Key Challenge	Free-tier caps	Two-step publish	Spam heuristics	App review

VI. PRIVACY AND AUTHENTICATION

A. OAuth 2.0 and Delegated Authorisation

OAuth 2.0 enables a third-party application to obtain scoped, time-limited access to a user's account on an external service without ever receiving that user's password [26]. The Authorization Code grant with PKCE is the recommended flow for public clients such as desktop or web applications, as it mitigates interception attacks on the authorisation code without requiring a client secret.

B. Google OAuth 2.0 as Identity Provider

Google OAuth 2.0 combines standard OAuth 2.0 authorisation with OpenID Connect (OIDC) identity tokens, pro-viding both access control and verified identity from a single flow [28]. For AutoPost-MCP, this offers three practical benefits: users authenticate with a provider they already trust, the application never stores passwords, and a mature infrastructure—including built-in multi-factor authentication and session anomaly detection—guards the authentication boundary on Google's behalf.

C. Token Security and Minimal Scope

After the Google OAuth flow, per-platform access tokens and refresh tokens are stored using the Python keyring library, which delegates to the host operating system's credential store. The system requests only the minimum OAuth scopes required for publishing (tweet.write, instagram_content_publish, w_member_social, pages_manage_posts). Refresh token rotation is handled transparently by each MCP server, eliminating the need for users to re-authenticate on every session.

VII. EXISTING TOOLS AND THEIR LIMITATIONS

A. Commercial Scheduling Platforms

Hootsuite and Buffer both provide calendar-based scheduling across multiple social networks [29]. Neither generates post content using AI; every message must be authored manually. Both platforms operate proprietary integrations with each social API, offering no open standard or programmatic extensibility. Hootsuite's entry-level pricing exceeds $99 per month, effectively excluding individual developers and student projects from evaluation or adoption.

B. Rule-Based Workflow Tools

Zapier and Make (formerly Integromat) enable conditional triggers between web services [30]. Connecting an LLM to a Zapier workflow requires chaining multiple third-party service connections, each representing an additional data-transit hop outside the user's control. These tools are also incapable of per-platform content adaptation; any content flowing through a Zap is identical across all destination channels.

C. Open-Source Research Prototypes

LangChain-based agent prototypes have demonstrated the technical feasibility of autonomous social media posting [23]. However, these implementations typically address a single platform, use plaintext credential storage, and integrate with social APIs via ad hoc tool definitions that cannot be shared or reused across projects. None employs MCP.

TABLE III: Feature Gap Analysis: Existing Tools vs. AutoPost-MCP

Feature	Hootsuite	Buffer	Zapier	Research	Proposed
AI Content Gen.	No	No	No	Partial	Yes
Multi-Platform	Yes	Yes	Yes	No	Yes
MCP Integration	No	No	No	No	Yes
Google OAuth	No	No	No	No	Yes
Platform Adaptation	No	No	No	No	Yes
Open-Source Core	No	No	No	Yes	Yes

VIII. PROPOSED AUTOPOST-MCP SYSTEM

A. System Overview

AutoPost-MCP is a Python-based automation framework in which a user: (1) authenticates once via Google OAuth 2.0;

(2) enters a natural-language content intent; (3) reviews four AI-generated, platform-adapted post variants; (4) approves or edits them; and (5) publishes to any combination of X, Instagram, LinkedIn, and Facebook through a single command, with all API communication handled by MCP servers operating behind the scenes.

B. Technology Stack

TABLE IV: AutoPost-MCP Technology Stack

Component	Technology
Language	Python 3.11+
Agent Framework	LangChain + LangGraph
LLM Backend	Claude (Anthropic API) / GPT-4o
Protocol	Anthropic MCP Python SDK
Authentication	google-auth + google-auth-oauthlib
X Integration	Tweepy \ (\rightarrow \) MCP Server
Instagram Integration	requests + Facebook SDK \ (\rightarrow \) MCP Server
LinkedIn Integration	requests + LinkedIn REST \ (\rightarrow \) MCP Server
Facebook Integration	facebook-sdk \ (\rightarrow \) MCP Server
Token Storage	Python keyring
Interface	Click CLI / FastAPI

C. Workflow Summary

The end-to-end workflow maps to the agent graph in Fig. 2. The Orchestrator receives intent and target-platform selections. The Content Agent runs one ReAct pass per platform, consulting platform-specific system prompts and any semantic memory in scope. Generated variants are surfaced for user review; rejection triggers a refined generation pass. On approval, the Publisher Agent dispatches MCP tool calls in parallel, collects responses, and writes a posting record to episodic memory.

IX. CHALLENGES AND ETHICAL CONSIDERATIONS

A. Technical Challenges

Rate limits across the four platforms are asynchronous and independently capped, requiring the Publisher Agent to implement per-platform back-off queues rather than a single global throttle. Instagram's mandatory media attachment creates a workflow divergence: purely text-based intents must either have an image generated or associated before the Instagram variant can be published. Platform API policies and endpoint schemas change frequently; the MCP server abstraction layer isolates these changes from the agent logic, but the servers themselves require ongoing maintenance.

B. Ethical Considerations

Automated posting at scale raises legitimate concerns about the authenticity of online discourse. Undisclosed AI-generated content may mislead audiences who assume they are reading a human author. The human-in-the-loop approval step is a structural mitigation: no content is published without explicit user confirmation. Additionally, because LLMs can reproduce and amplify biases present in their training data, the Content Agent's outputs should be subjected to bias-checking tooling before deployment at significant scale. Finally, automated posting should comply with each platform's terms of service regarding bot disclosure and frequency limits.

X. RESEARCH GAPS AND FUTURE DIRECTIONS

A. Identified Gaps

No published system applies MCP as the integration back-bone for cross-platform social media automation; existing literature treats MCP primarily in the context of coding assistants and data-retrieval agents. Deep, empirically validated per-platform content adaptation strategies remain under-studied: most research reports qualitative assessments rather than engagement-rate benchmarks. Privacy-by-design principles are largely absent from open-source automation prototypes, which typically store credentials in plaintext. Finally, there is no established benchmark suite for evaluating the content quality, publishing reliability, or latency profile of agent-based social media automation systems.

B. Future Work

Planned extensions to AutoPost-MCP include:

- Engagement-aware scheduling:

A Scheduler Agent that queries platform analytics APIs to identify audience-active time windows and queue posts accordingly.

- Visual content generation:

An image-generation MCP server (DALL-E or Stable

Diffusion) that produces Instagram-ready visuals from the caption context.

- Multilingual support:

Extending the Content Agent to generate posts in regional languages based on audience demographics.

- Closed-loop learning:

Feeding post-performance metrics back into semantic memory so that the Content Agent progressively refines style towards higher-engagement outputs.

- Platform extension:

Adding YouTube (YouTube Data API), Threads, and Pinterest as additional MCP servers.

- Benchmark development:

Creating a publicly available evaluation framework covering content naturalness, AI detectability scores, publishing latency, and rate-limit compliance.

XI. CONCLUSION

This review has examined the technological and research foundations of AutoPost-MCP: an AI-driven, agent-based, cross-platform social media automation system built around the Model Context Protocol. The survey spanned social media automation history, LLM content generation techniques, multi-agent system patterns, the MCP specification and its rapid 2025 ecosystem growth, four major platform APIs, and privacy-preserving authentication via Google OAuth 2.0.

The analysis reveals that existing tools address individual aspects of the cross-platform automation problem but leave significant gaps: no tool combines MCP-native integration with per-platform AI content adaptation and privacy-first authentication. AutoPost-MCP's layered architecture, Orchestrator–Content–Publisher agent decomposition, and per-platform prompting strategy are each grounded in surveyed literature and designed to address these gaps directly.

Key contributions of this review are: a synthesised rationale for MCP as a social media integration backbone; a formalised multi-agent workflow pattern for content automation pipelines; and a gap analysis situating the proposed system against the state of the art. Future work will centre on empirical evaluation, engagement-aware scheduling, visual content generation, and the development of a community benchmark for agent-based social media systems.

ACKNOWLEDGEMENTS

The authors thank their project supervisor for guidance throughout this work and acknowledge the open-source com-munities behind LangChain, LangGraph, the Anthropic MCP SDK, and the social media developer programmes that make this research possible.

REFERENCES

[1] Statista Research Department, "Number of social media users worldwide from 2017 to 2025," *Statista*, 2025.
[2] DataReportal, "Digital 2026: Global Overview Report," Oct. 2025.
[3] Anthropic, "Introducing the Model Context Protocol," Anthropic Blog, Nov. 2024.
[4] Model Context Protocol Maintainers, "One Year of MCP: November 2025 Specification Release," Nov. 2025.
[5] Thoughtworks, "The Model Context Protocol's impact on 2025," Thoughtworks Insights, Dec. 2025.
[6] T. B. Brown et al., "Language models are few-shot learners," in Proc. NeurIPS, vol. 33, 2020, pp. 1877–1901.
[7] A. Vaswani et al., "Attention is all you need," in Proc. NeurIPS, vol. 30, 2017, pp. 5998–6008.
[8] J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," in Proc. NeurIPS, vol. 35, 2022, pp. 24824–24837.
[9] L. Ouyang et al., "Training language models to follow instructions with human feedback," in Proc. NeurIPS, vol. 35, 2022, pp. 27730–27744.
[10] H. Touvron et al., "LLaMA: Open and efficient foundation language models," arXiv preprint arXiv:2302.13971, 2023.
[11] W. Hua et al., "WarAgent: Large language model-based multi-agent simulation of world wars," arXiv preprint arXiv:2311.17227, 2023.
[12] A. Uchendu et al., "TURINGBENCH: A benchmark for Turing test in the age of neural text generation," in Findings of EMNLP, 2021.
[13] E. Mitchell et al., "DetectGPT: Zero-shot

machine-generated text detection using probability curvature," in Proc. ICML, 2023.
[14] B. Guo et al., "How close is ChatGPT to human experts? Comparison corpus, evaluation, and detection," arXiv preprint arXiv:2301.07597, 2023.
[15] J. S. Park et al., "Generative agents: Interactive simulacra of human behavior," in Proc. UIST, 2023.
[16] S. Hong et al., "MetaGPT: Meta programming for multi-agent collaborative framework," arXiv preprint arXiv:2308.00352, 2023.
[17] M. Wooldridge and N. R. Jennings, "Intelligent agents: Theory and practice," Knowl. Eng. Rev., vol. 10, no. 2, pp. 115–152, 1995.
[18] Q. Wu et al., "AutoGen: Enabling next-generation LLM applications via multi-agent conversation," arXiv preprint arXiv:2308.08155, 2023.
[19] G. Li et al., "CAMEL: Communicative agents for 'mind' exploration of large language model society," in Proc. NeurIPS, 2023.
[20] S. Yao et al., "ReAct: Synergizing reasoning and acting in language models," arXiv preprint arXiv:2210.03629, 2022.
[21] N. Shinn et al., "Reflexion: Language agents with verbal reinforcement learning," in Proc. NeurIPS, 2023.
[22] T. Schick et al., "Toolformer: Language models can teach themselves to use tools," in Proc. NeurIPS, 2023.
[23] H. Chase, "LangChain: Building applications with LLMs through composability," GitHub, 2023.
[24] Significant Gravitas, "Auto-GPT: An autonomous GPT-4 experiment," GitHub Repository, 2023.
[25] C. Cadwalladr and E. Graham-Harrison, "Cambridge Analytica data breach," The Guardian, Mar. 2018.
[26] D. Hardt, "The OAuth 2.0 Authorization Framework," IETF RFC 6749, Oct. 2012.
[27] T. Lodderstedt et al., "OAuth 2.0 Security Best Current Practice," IETF RFC 9700, 2025.
[28] Google LLC, "Using OAuth 2.0 to Access Google APIs," Google Identity Documentation, 2024.
[29] Hootsuite Inc., "Hootsuite: Social Media Management Platform," 2023.
[30] Zapier Inc., "Zapier: Automation for busy people," 2023.
[31] X Corp., "X API v2 Documentation," 2024–2026.
[32] Meta Platforms Inc., "Instagram Graph API Reference," 2024–2026.
[33] LinkedIn Corporation, "LinkedIn API Documentation," 2024–2026.
[34] Meta Platforms Inc., "Facebook Graph API Reference," 2024–2026.