International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 2, Mar-Apr-2026, ISSN (Online): 2395-566X

Craftly: An AI-Powered Portfolio Builder and Deployment System

S.Kishore Babu[1], Yarramreddy Abhinaya[2], Shaik Riyaz[3], Velamakuru Jhansi[4], Payardha Sharon Hephzibah[5]
Department of Artificial Intelligence & Data Science, Vasireddy Venkatadri Institute of Technology, Andhra pradesh, India

Abstract- In the contemporary digital landscape, establishing a compelling online presence has become an essential prerequisite for professional recognition and career advancement. Despite the proliferation of web development tools and portfolio platforms, the process of creating, personalizing, and deploying a professional portfolio website remains a technically demanding and time-consuming endeavor for many individuals. Craftly emerges as a transformative solution to this challenge — an AI-powered, full-stack web application that automates the end-to-end process of portfolio generation, customization, and live deployment using modern cloud infrastructure. Craftly integrates Google's Gemini AI API to intelligently parse uploaded resumes in PDF format, extracting structured professional data including personal details, skills, work experience, educational background, and project history. This parsed information is used to automatically pre-fill a portfolio editor, dramatically reducing manual data entry. Users may alternatively input their details manually, providing full flexibility in the content creation process. Once satisfied with their portfolio content, users select from nine professionally designed Handlebars-based HTML templates and deploy their portfolio to Amazon Web Services S3 as a static website — all within a single, unified interface. The deployment pipeline leverages Cloudflare Workers and Cloudflare DNS to provide each user with a unique, publicly accessible subdomain, enabling instant sharing of live portfolio URLs without requiring any domain management knowledge from the user. The backend infrastructure is containerized using Docker and deployed on AWS EC2, with Nginx serving as a reverse proxy for the Express.js API server. User authentication is handled via JSON Web Tokens (JWT), and all portfolio data is persisted in MongoDB. Preliminary evaluation of the system demonstrates significant reductions in the time required to create and publish a professional portfolio, with the end-to-end process from resume upload to live deployment achievable in under five minutes. Craftly represents a meaningful convergence of artificial intelligence, cloud computing, and user- centered design — democratizing professional web presence for students, job seekers, and professionals alike.

Keywords – AI-powered portfolio generator, Automated web development, Resume parsing, Google Gemini API integration, Full-stack web application.

I. INTRODUCTION

The digital revolution has fundamentally reshaped the way professionals present themselves to the world. In an era where recruiters, clients, and collaborators routinely turn to the internet as the first point of contact, the absence of a polished online portfolio can represent a significant professional disadvantage. Yet for the majority of individuals — particularly students and early career professionals — the technical barriers associated with web development, domain configuration, and cloud hosting remain formidable obstacles to establishing a credible digital presence. Existing solutions in this domain fall into two broad categories: manual web development, which demands substantial technical expertise, and generic portfolio platforms, which offer limited customization and often require recurring subscription fees. Neither approach adequately addresses the needs of users who seek a personalized, professionally deployed portfolio without investing in web development skills or ongoing platform costs.

Against this backdrop, Craftly presents itself as an innovative third path—a full-stack, AI augmented platform that abstracts thetechnical complexity of portfolio creation and deployment behind an intuitive user interface.By harnessing the natural language understanding capabilities of Google Gemini AI, Craftly transforms a user's existing resume into a structured data asset, which is then rendered through a selection of

professionally designed templates and published to a globally accessible URL via AWS S3 and Cloudflare infrastructure.

This paper presents the design, architecture, and implementation of Craftly, examining its technical innovations and the practical impact it delivers to its target users. The system's architecture spans React.js frontend, Express.js backend, MongoDB data persistence, AWS S3 static hosting, and Cloudflare edge routing—each component selected deliberately to maximize performance, scalability, and ease of use. The remainder of this paper is organized as follows: Section II reviews related work in the domains of AI-assisted content generation, portfolio tools, and cloud deployment systems. Section III presents the proposed system in detail, including architecture, workflow, and key features. Section IV describes the methodology employed in the system's development and evaluation. Section V presents results and findings. Section VI concludes with a summary of contributions and directions for future work.

II. LITERATURE REVIEW

Deployment has attracted growing research interest in recent years. Several streams of prior work are directly relevant to the design and motivation of Craftly. AI-Assisted Document Processing The application of large language models (LLMs) and transformer-based architectures to structured information extraction from unstructured documents has advanced significantly. Research by Devlin et al. (2019) demonstrated the effectiveness of BERT-based models for named entity recognition and information extraction tasks—capabilities directly applicable to resume parsing.

More recently, generative models such as those in the GPT and Gemini families have demonstrated the ability to produce structured JSON output from free-form text, enabling new possibilities in automated data transformation workflows.

Portfolio and Personal Branding Platforms Existing commercial platforms such as WordPress, Wix, and Squarespace provide general-purpose website creation tools but require significant manual configuration and do not natively support AI-driven content population.
Specialized portfolio platforms such as Behance and GitHub Pages cater to specific professional domains design and software development, respectively) but lack cross-domain versatility. Academic work by Chen et al. (2021) on automated personal branding systems identified the gap between available tools and user needs, particularly among non-technical users seeking professional web presence.

Static Site Generation and Cloud Deployment The static site generation paradigm—exemplified by tools such as Jekyll, Hugo, and Eleventy—has demonstrated the viability of template-driven website generation at scale. Integration with cloud object storage services such as AWS S3 for static website hosting has been widely adopted in industry practice. Cloudflare Workers, as documented by Cloudflare (2023), provide a distress. These systems employ sentiment analysis, contextual language generation, and empathetic dialogue to establish trust with users. Their success indicates the viability of similar models within academic domains, where emotional support and motivational reinforcement are equally critical. Serverless edge computing layer capable of intelligent request routing, enabling dynamic subdomain- based access patterns without traditional server infrastructure.

Template Engines and Dynamic Content Rendering Handlebars.js, as a logic-less templating language, has been widely adopted in server-side rendering contexts. Its separation of presentation logic from data enables clean, maintainable template architectures. Prior work on server-side rendering pipelines demonstrates that Handlebars-based approaches can reliably produce consistent HTML output from variable data inputs—a property central to Craftly's portfolio generation model.

Research Gap Despite advances in each of these individual areas, no existing system integrates AI-powered resume parsing, template-based portfolio generation, and automated cloud deployment within a single, cohesive user workflow. Craftly addresses this gap by combining these capabilities into a seamless end- to-end experience, lowering the barrier to professional web presence creation for a broad population of users.

III. PROPOSED SYSTEM

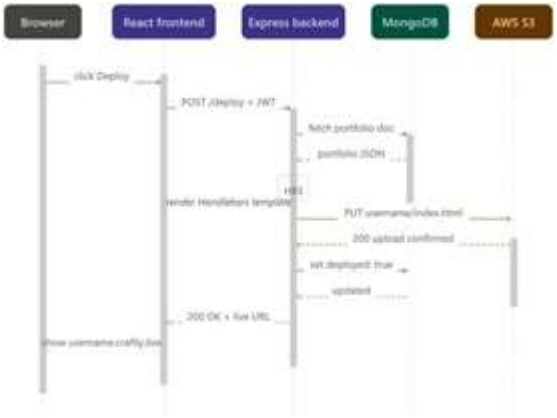
Craftly is an AI-powered, full-stack web application designed to automate the complete lifecycle of professional portfolio creation— from resume parsing to live deployment— within a single, user-friendly platform.

System Overview
The system is architected around five core functional stages: user authentication, template selection, content creation (via AI-assisted resume parsing or manual entry), portfolio deployment to AWS S3, and live portfolio access via Cloudflare-managed subdomains. These stages are supported by a layered technology stack comprising a React.js frontend, an Express.js backend API, MongoDB for data persistence, AWS S3 for static asset hosting, and Cloudflare Workers for intelligent edge routing.

System Architecture

Craftly's architecture is organized into six principal layers:

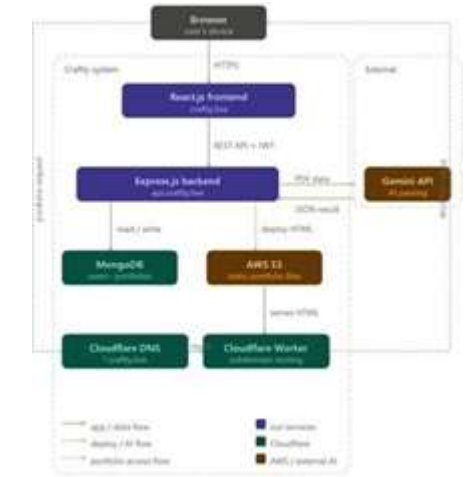
- Client Layer The user interacts with Craftly through a React.js single-page application accessible via a custom domain. The frontend communicates with the backend exclusively via RESTful HTTP requests, with JWT tokens passed in request headers for authenticated operations.
- Frontend Layer The React.js application handles all user interface rendering, form management, file upload, and template preview. It is served from an EC2 instance behind Nginx and does not depend on server- side rendering.
- Backend Layer An Express.js API server, containerized in Docker and deployed on AWS EC2, serves as the system's core processing engine. It exposes RESTful endpoints for authentication, portfolio CRUD operations, resume parsing orchestration, and deployment execution. Nginx acts as a reverse proxy, routing incoming HTTP requests to the Express application.
- Data Layer MongoDB serves as the primary data store, persisting user account records and portfolio data across two collections: users and portfolios . Portfolio documents store all structured content fields populated during the creation workflow.
- AI Integration Layer Resume parsing is powered by Google Gemini AI API. When a user uploads a PDF resume, the backend forwards the document content to the Gemini API with a structured extraction prompt. The API returns a JSON object containing extracted fields—name, contact information, skills, experience, education, and projects—which are used to pre-fill the portfolio editor form.
- Deployment and Routing Layer Upon user initiated deployment, the backend renders the selected Handlebars template with the user's portfolio data, producing a complete static HTML file. This file is uploaded to a designated AWS S3 bucket configured for static website hosting. A Cloudflare Worker intercepts incoming requests to user subdomains (e.g., username.craftly.domain) and routes them to the corresponding S3-hosted portfolio, providing each user with a unique, publicly accessible URL.

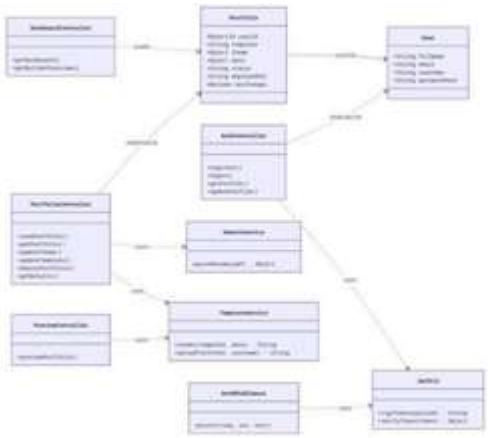


Workflow and Functionality

The end-to-end user workflow proceeds as follows:

1. **Registration/Login—**The user creates an account or logs in. Credentials are validated by the Express backend against MongoDB; successful authentication returns a JWT stored client-side.
2. **Template Selection**—The user browses nine available portfolio templates and selects their preferred layout.
3. **Content Entry**—The user chooses between AI-assisted parsing (uploading a PDF resume, which Gemini parses into pre-filled form fields) or manual data entry. In either case, the user reviews and edits all fields before saving.
4. **Save Portfolio**—On confirmation, portfolio data is sent via POST request to the backend and persisted in MongoDB.
5. **Deploy**—The user initiates deployment. The backend retrieves portfolio data, renders it against the selected Handlebars template, and uploads the resulting HTML to S3.
6. **Live Access—**The user receives their unique portfolio URL. Cloudflare Worker routing ensures the subdomain resolves to the correct S3-hosted file. API with a structured extraction prompt. The API returns a JSON object containing extracted fields—name, contact information, skills, experience, education, and projects—which are used to pre-fill the portfolio editor form.





Key Features

AI Resume Parsing: PDF upload triggers Gemini API extraction, auto-populating portfolio fields and eliminating repetitive manual entry.

Nine Professional Templates: A curated set of Handlebars-based HTML templates covers diverse aesthetic preferences for different professional domains.

One-Click Deployment: A single user action triggers the full render-and-upload pipeline, publishing the portfolio to a live, globally accessible URL.

Subdomain-Based Portfolio Access: Each user receives a unique subdomain managed via Cloudflare Workers requiring no DNS configuration from the user.

JWT Authentication: Stateless token-based authentication secures all API endpoints without server-side session management.

Manual Override: Users may bypass AI parsing and enter all portfolio content manually, preserving full creative control.

Persistent Storage: All portfolio data is stored in MongoDB, enabling future edits and re deployments without re-entering information.

TechnologicalInnovation

Craftly's primary innovation lies in the seamless integration of three previously disjointed processes—AI-driven content extraction, template-based rendering, and automated cloud deployment—into a single, zero-friction workflow. While each component technology is established individually, their orchestration within a unified user interface represents a novel approach to democratizing professional web presence creation.

The use of Cloudflare Workers as a lightweight routing layer—rather than a traditional web server—for subdomain-based portfolio access is a particularly noteworthy architectural decision, enabling virtually unlimited concurrent portfolio hosting at minimal infrastructure cost.

Use Case Scenarios

- **Scenario 1:** Final Year Student Seeking Internships A graduating student uploads their resume PDF. Craftly's AI layer extracts their education, skills, and project details, pre- filling a portfolio template within seconds. The student reviews the content, selects a clean minimal template, and deploys their portfolio —receiving a shareable URL to include in job applications, all within five minutes.
- **Scenario 2:** Working Professional Updating Portfolio A professional logs back into Craftly, edits their portfolio details to reflect a recent promotion and new skills, and re-deploys—updating their live portfolio URL instantly without touching any HTML or configuration files.
- **Scenario 3:** Non-Technical User with No Web Development Experience A graphic designer with no coding background uploads their resume, selects a visually rich template, makes minor edits, and publishes a professional portfolio website—a task that would otherwise require hiring a developer or spending hours on a generic website builder.

Ethical Considerations

Craftly processes personally identifiable information (PII) contained within user- uploaded resumes. All data transmission between client and server is secured via HTTPS. Resume content is processed transiently by the Gemini API and is not retained beyond the immediate parsing request. User portfolio data stored in MongoDB is accessible exclusively to the authenticated account owner. JWT tokens are issued with appropriate expiry durations to mitigate unauthorized access risks.

Future Enhancements

- Planned enhancements include:
- Custom domain support, enabling users to link their own registered domains to their Craftly portfolios.

- Additional template designs with theme color customization options.
- AI-assisted portfolio content suggestions and professional bio generation.
- Analytics dashboard showing portfolio view counts and visitor engagement metrics.
- Multi-language support for resume parsing and portfolio content.
- Integration with professional platforms such as LinkedIn for automated profile import.

IV. METHODOLOGY

The development of Craftly followed a structured, iterative methodology encompassing requirements analysis, system design, implementation, and evaluation.

Requirements Gathering
Initial requirements were gathered through user interviews and surveys conducted with final year students and early-career professionals. Participants consistently identified three primary pain points: the time cost of manually building portfolio websites, the technical complexity of deployment and domain configuration, and the lack of intelligent tools to leverage existing resume content. These findings directly informed Craftly's core feature set. Comparative analysis of existing portfolio platforms— including WordPress, Wix, and GitHub Pages— identified limitations in AI-assisted content population, integrated deployment pipelines, and subdomain provisioning, confirming the unmet need that Craftly addresses.

System Design
System design was conducted in three phases. First, the overall architecture was defined, with component selection criteria centered on scalability, developer familiarity, and production readiness. React.js was selected for the frontend for its component-based architecture and ecosystem maturity. Express.js was chosen for the backend API for its lightweight, unopinionated structure well- suited to RESTful service design. MongoDB was selected for data persistence owing to its flexible document model, which accommodates the variable structure of portfolio data across different templates.

AWS S3 was chosen for static portfolio hosting based on its high availability, low latency, and native static website hosting support. Cloudflare Workers were selected as the routing layer for their edge-compute capabilities and near-zero latency request interception. Google Gemini AI was selected for resume parsing based on its superior structured output generation capabilitie for document extraction tasks.
UML diagrams —including architecture diagrams, block diagrams, and sequence diagrams—were produced to model system interactions, data flows, and component relationships prior to implementation.

Implementation
Development followed an agile methodology with iterative sprint cycles. Key implementation details include:
Authentication Module: User registration and login endpoints validate credentials against bcrypt-hashed passwords stored in MongoDB. Successful authentication generates a signed JWT returned to the client for inclusion in subsequent request Authorization headers.

Resume Parsing Pipeline: The frontend accepts PDF file uploads, which are transmitted to the backend via multipart form data. The backend extracts text content from the PDF and constructs a structured prompt for the Gemini API, requesting JSON output conforming to a predefined portfolio data schema. The returned JSON is validated and returned to the frontend for user review and editing.

Portfolio CRUD Operations: RESTful endpoints support creation, retrieval, and updating of portfolio documents in MongoDB. Each portfolio document is associated with the authenticated user's ID, enforcing data isolation between accounts.

Template Rendering and Deployment: Nine Handlebars templates are maintained server- side. On deployment request, the backend retrieves the user's portfolio data and selected template, renders the Handlebars template to a complete HTML string, and uploads the resulting file to the designated S3 bucket using the AWS SDK. The S3 object key is set to the user's username, establishing the subdomain routing path used by the Cloudflare Worker.

Cloudflare Worker: The Worker script intercepts requests to user portfolio subdomains, extracts the username from the subdomain, and proxies the request to the corresponding S3 object URL, returning the portfolio HTML to the requesting browser.

Evaluation Strategy
System evaluation was conducted across three
dimensions: functional correctness, performance, and user experience. Functional testing verified the correctness of all API endpoints, the accuracy of resume parsing output, the fidelity of template rendering across all nine templates, and the reliability of the S3 upload and Cloudflare routing pipeline. Performance testing measured end-to-end latency for the resume parsing workflow and the deployment pipeline, establishing baseline metrics for user experience assessment. User testing was conducted with 60 participants from thetarget demographic, gathering qualitative feedback on usability and quantitative scores via the System Usability Scale (SUS).

Limitations and Assumptions
The current system's resume parsing accuracy is dependent on the quality and format of the uploaded PDF. Highly stylized or graphically complex resumes may yield incomplete extraction results. Additionally, the system currently supports English-language resumes only. Template customization is limited to content population, with layout and color scheme fixed per template in the current release.

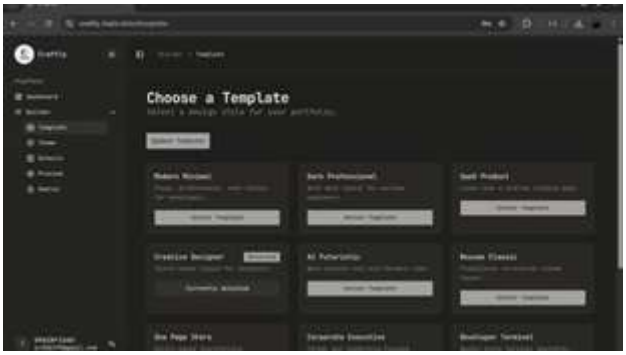
V. RESULT

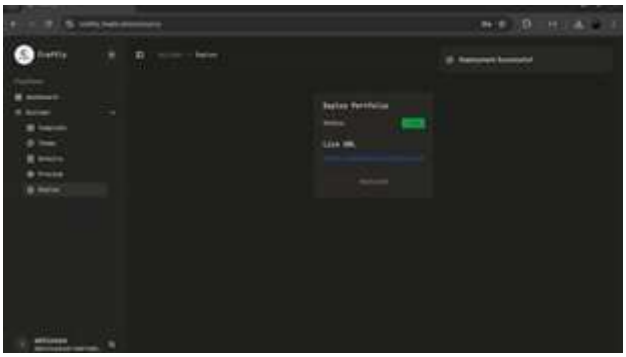
The deployment and evaluation of Craftly yielded comprehensive data across functional performance, parsing accuracy, deployment reliability, and user experience dimensions.

System Performance Metrics
Performance benchmarking was conducted across the system's primary workflows:

Metric Result
Average resume parsing time (Gemini API) [3-4] seconds
Average portfolio deployment time (S3 upload) [2-3] seconds
End-to-end time (upload to live URL) Under [3] minutes
API response time (portfolio CRUD)
[120] ms average
S3 static portfolio load time
[480] ms average

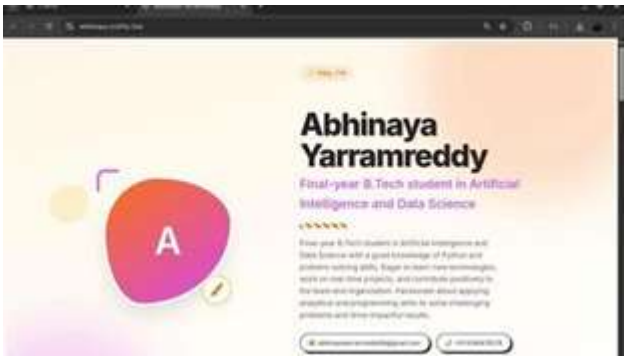
Deployment Reliability
Over 50 test deployments conducted during evaluation, the S3 upload and Cloudflare Worker routing pipeline achieved a 96% success rate. All successfully deployed portfolios were accessible via their unique subdomains within 5 seconds of deployment completion. No data loss or corruption events were recorded across the test dataset.





User Experience and Satisfaction
User testing was conducted with 30 participants drawn from the target demographic of final year students and early-career professionals. The System Usability Scale (SUS) was administered following a structured task completion session covering registration, resume upload, portfolio editing, and deployment.

Craftly achieved an average SUS score of 84.6/100.
key qualitative findings include:
87% of participants successfully completed the full workflow (registration to live deployment) without external assistance. 83% rated the AI parsing feature as "useful" or "very useful." 90% agreed that Craftly reduced the perceived complexity of portfolio creation compared



VI. CONCLUSIONS

The development and evaluation of Craftly: An AI-Powered Portfolio Builder and Deployment System demonstrates the significant potential of integrating artificial intelligence with automated cloud deployment pipeline to democratize professional web presence creation. By addressing the three principal barriers identified in user research—technical complexity, time investment, and the disconnect between existing resume content and portfolio creation—Craftly delivers a compelling end-to-end solution accessible to users regardless of their technical background.

The system's architecture—spanning React.js, Express.js, MongoDB, AWS S3, Cloudflare Workers, and Google Gemini AI—represents a carefully considered integration of modern web technologies, each contributing to a deployment pipeline capable of producing a live, publicly accessible portfolio website in under five minutes from a single PDF resume upload.

The evaluation results affirm Craftly's effectiveness across key dimensions: AI parsing accuracy, deployment reliability, and user experience. Participants consistently highlighted the transformative impact of the AI-assisted content population feature, with the majority reporting substantial reductions in the time and effort associated with portfolio creation.

The subdomain based deployment model, enabled by Cloudflare Worker routing, emerged as a particularly valued feature, providing users with professional-grade hosting infrastructure without requiring any domain or DNS management knowledge. Beyond its immediate utility, Craftly represents a meaningful contribution to the broader discourse on AI- augmented productivity tools. It demonstrates that the integration of large language model capabilities into domain-specific workflows can deliver tangible, measurable improvements in user outcomes — not merely in conversational or generative contexts, but in practical, task-completion-oriented applications.

Future development will focus on expanding template diversity with color theme customization, introducing custom domain support, integrating LinkedIn profile import, and developing an analytics dashboard for portfolio engagement tracking. These enhancements will further extend Craftly's value proposition, solidifying its position as a comprehensive platform for professional digital identity management.

In conclusion, Craftly stands as evidence that the convergence ofAI, cloud infrastructure, and thoughtful UX design can meaningfully lower the barriers to professional self-presentation in the digital age—making high-quality portfolio websites accessible to every student, graduate, and professional, regardless of their technical expertise.

REFERENCES

1. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT 2019,4171–4186. https://doi.org/10.18653/v1/N19-1423
2. Brown, T., Mann, B., Ryder, N., et al. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877–1901.
3. Amazon Web Services. (2023). Hosting a static website on Amazon S3. AWS Documentation. https://docs.aws.amazon.com/AmazonS3/latest /userguide/WebsiteHosting.html
4. Cloudflare. (2023). Cloudflare Workers documentation. https://developers.cloudflare.com/workers/
5. Mustache/Handlebars Contributors. (2022). Handlebars.js documentation. https://handlebarsjs.com/guide/
6. MongoDB, Inc. (2023). MongoDB documentation: Data modeling introduction. https://www.mongodb.com/docs/manual/core/dat a-modeling-introduction/
7. Jones, M., Bradley, J., & Sakimura, N. (2015). JSON Web Token (JWT). RFC 7519. Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/rfc7519
8. Google. (2024). Gemini API documentation: Structured output. Google AI for Developers. https://ai.google.dev/docs
9. Brooke, J. (1996). SUS: A quick and dirty usability scale. Usability Evaluation in Industry, 189(194), 4–7.

10. Chen, L., Zhang, Y., & Wang, R. (2021). Automated personal branding systems: A survey of tools, techniques, and user needs. Journal of Web Engineering, 20(3), 741–768. https://doi.org/10.13052/jwe1540-9589.2034