TẠP CHÍ KHOA HỌC TRƯỜNG ĐẠI HỌC BẠC LIÊU
Bac Lieu University Journal of Science
Lĩnh vực Khoa học công nghệ
ISSN 2734-973X | e-ISSN 3093-3420

SAFE NEWS: TIẾP CẬN AI TẠO SINH TRONG TỰ ĐỘNG HÓA LỌC TIN VÀ GIẢM THIỂU TÁC ĐỘNG TIÊU CỰC
SAFE NEWS: A GENERATIVE AI APPROACH FOR AUTOMATED FILTERING AND MITIGATING NEGATIVE NEWS IMPACT

Phạm Yến Nhi*, Nguyễn Trần Anh Khoa, Nguyễn Thúy Anh, Lê Anh Tuấn

Khoa Công nghệ thông tin - Trường Đại học Kỹ thuật – Công nghệ Cần Thơ
*pynhi@ctuet.edu.vn

Ngày nhận bài: 03/3/2026
Ngày chấp nhận đăng: 05/5/2026

Keywords: *Doomscrolling, Generative AI, Semantic Analysis, Software Architecture.*

Từ khóa: *Doomscrolling, Generative AI, Semantic Analysis, Software Architecture.*

ABSTRACT

The proliferation of digital information has exacerbated "doomscrolling," negatively impacting user mental health, while traditional keyword-based filtering remains insufficient for addressing semantic nuances. This study proposes "Safe News," a system integrating Generative AI within a modern layered software architecture to automate content moderation. We formulate a decision function based on semantic feature vectors extracted via Large Language Models, combined with a Gamification algorithm tracking reading sessions to promote positive user habits. Experimental evaluation on test datasets demonstrates a classification accuracy of 80% with an average processing latency of 2.1s to 2.5s, satisfying real-time requirements. The research confirms the feasibility of integrating quantitative mathematical models with the inference capabilities of Generative AI to address information processing challenges, establishing a technical foundation for next-generation safe content recommendation systems.

TÓM TẮT

Sự bùng nổ của thông tin số đã làm gia tăng hiện tượng "doomscrolling", gây ảnh hưởng tiêu cực đến sức khỏe tinh thần người dùng, trong khi các bộ lọc từ khóa truyền thống bộc lộ hạn chế lớn về khả năng xử lý ngữ nghĩa . Nghiên cứu này đề xuất "Safe News" – một hệ thống tích hợp Trí tuệ nhân tạo tạo sinh nhằm tự động hóa quy trình kiểm duyệt nội dung. Đóng góp chính của nghiên cứu này là đề xuất một khung kiến trúc phần mềm mới (Software Architecture) tích hợp AI tạo sinh để tự động hóa quy trình lọc tin dựa trên nhận thức ngữ cảnh sâu. Chúng tôi hiện thực hóa hàm quyết định lọc dựa trên vectơ đặc trưng ngữ nghĩa được trích xuất từ mô hình ngôn ngữ lớn, kết hợp với thuật toán Gamification theo dõi phiên đọc để định hướng thói quen người dùng . Kết quả thực nghiệm trên tập dữ liệu kiểm thử cho thấy hệ thống đạt độ chính xác phân loại 80% với độ trễ xử lý trung bình từ 2.1s đến 2.5s, đáp ứng tốt yêu cầu thời gian thực. Kết quả nghiên cứu cho thấy của việc kết hợp mô hình toán học định lượng với năng lực suy luận của AI trong bài toán xây dựng hệ sinh thái tin tức an toàn.

1. GIỚI THIỆU

Trong kỷ nguyên số, thói quen tiếp cận thông tin của con người đã chuyển dịch mạnh mẽ từ báo giấy sang các nền tảng trực tuyến. Tuy nhiên, sự cạnh tranh về sự chú ý đã thúc đẩy các nền tảng ưu tiên hiển thị các tin tức giật gân, tiêu cực nhằm tối đa hóa thời gian tương tác của người dùng. Hiện tượng này dẫn đến hành vi "doomscrolling" – lướt liên tục qua các bản tin thảm họa, bạo lực hoặc gây lo âu (Shabahang và cộng sự, 2021).Các nghiên cứu gần đây trong giai đoạn 2023-2025 đã chỉ ra mối tương quan chặt chẽ giữa việc tiếp cận thường xuyên với tin tức tiêu cực và sự gia tăng các rối loạn lo âu, trầm cảm và căng thẳng trong cộng đồng (Nguyễn, 2020). Hơn thế nữa, các nghiên cứu chuyên sâu chỉ ra rằng doomscrolling không đơn thuần là sự thiếu hụt khả năng tự kiểm soát của cá nhân, mà là hệ quả trực tiếp từ cấu trúc thiết kế của các nền tảng số. Cơ chế này tạo ra một vòng lặp phản hồi mang tính cưỡng chế, khiến người dùng liên tục tiếp xúc với các nội dung kích động tâm lý dù nhận thức được tác hại (Dixit và cộng sự, 2025).

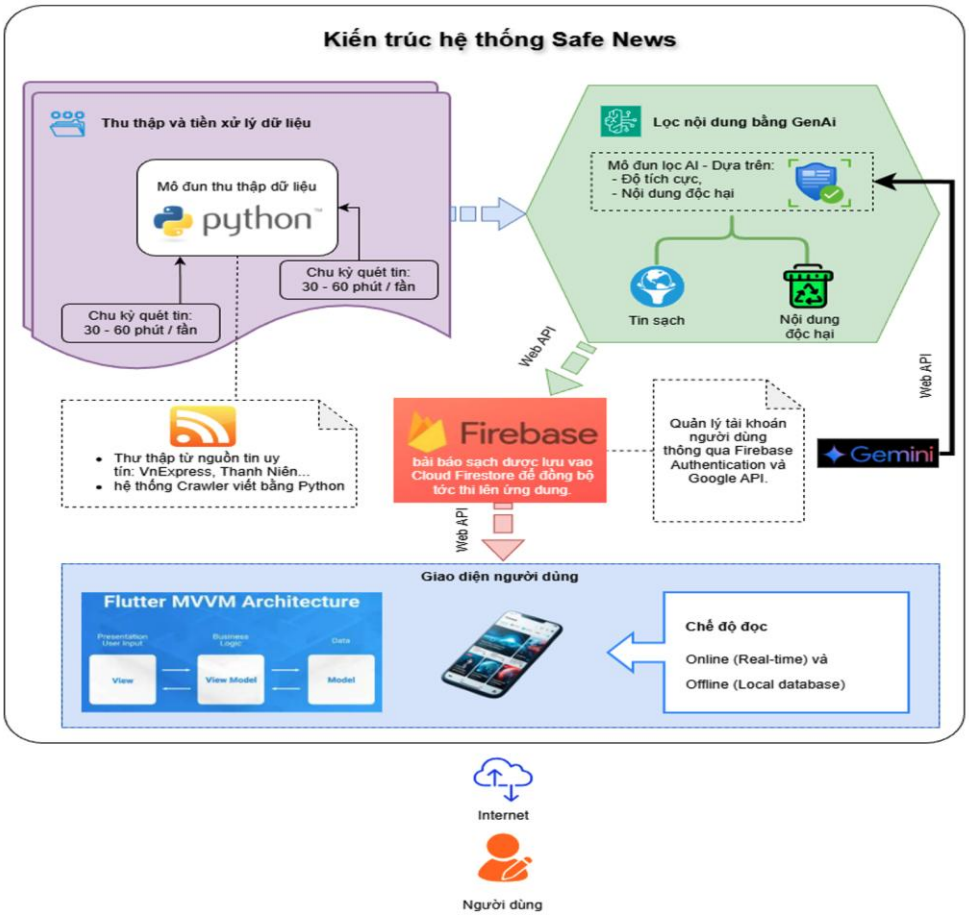
Việc kiểm duyệt nội dung thủ công không còn khả thi trước lượng dữ liệu khổng lồ được tạo ra mỗi ngày. Các phương pháp lọc truyền thống dựa trên từ khóa hoặc các mô hình học máy cổ điển (như SVM, Naive Bayes) thường gặp hạn chế trong việc hiểu ngữ cảnh và sắc thái ngôn ngữ (Fu và cộng sự, 2009). Sự hạn chế này bắt nguồn từ việc các mô hình học máy phân loại truyền thống phụ thuộc cốt lõi vào các kỹ thuật trích xuất đặc trưng thủ công và đo lường tần suất từ vựng, khiến chúng gần như "bất lực" trước các cấu trúc ngữ nghĩa phức tạp, sự mỉa mai, hoặc thông tin bị nhiễu bối cảnh. Sự ra đời của các Mô hình Ngôn ngữ Lớn (Large Language Models - LLMs) (Johnsen, 2024) như GPT-4 hay Gemini đã mở ra hướng đi mới cho bài toán này nhờ khả năng hiểu ngôn ngữ tự nhiên vượt trội. Thực tế, nghiên cứu của Gabriel và cộng sự (2024) đã chứng minh các can thiệp dựa trên LLMs có thể cải thiện độ chính xác của người dùng trong việc đánh giá độ tin cậy thông tin lên tới 47,6%, đặc biệt hiệu quả khi kết hợp với các giải thích cá nhân hóa. Tuy nhiên, đây là một công nghệ lưỡng dụng chứa đựng nhiều thách thức, Sandrini và Somogyi (2023) còn lo ngại rằng trong giai đoạn sơ khai, công nghệ này có thể gây nhiễu loạn hành vi tiêu dùng và suy giảm phúc lợi xã hội nếu thiếu các chiến lược kiểm soát phù hợp.

Trong bối cảnh hiện tại, Trí tuệ nhân tạo tạo sinh đang thể hiện hai vai trò đầy thách thức. Một mặt, công nghệ này làm trầm trọng thêm cuộc khủng hoảng thông tin khi bị lạm dụng để sản xuất nội dung nhiễu (giả) với chi phí gần bằng không, làm suy giảm niềm tin vào tin tức số (Campante và cộng sự, 2024). Nhưng mặt khác, chính năng lực suy luận tạo sinh này lại cung cấp chìa khóa để xây dựng các bộ lọc tự động hóa (Chang & Wang, 2025). Tổng hợp các nghiên cứu (Loth và cộng sự, 2024; Kieslich và cộng sự, 2025), khoảng trống nghiên cứu cốt lõi hiện nay không còn nằm ở việc chứng minh năng lực hiểu ngôn ngữ của LLM, mà nằm ở bài toán kiến trúc phần mềm: Làm thế nào để thiết kế một hệ thống có khả năng cô lập chi phí tính toán nặng nề của AI, đồng thời duy trì luồng trải nghiệm thời gian thực cho người dùng cuối để định hướng thói quen đọc lành mạnh thông qua Gamification.

Trong bài báo này, nhóm nghiên cứu tập trung giải quyết bài toán trên bằng cách đề xuất một kiến trúc mới "SafeNew", kết hợp sức mạnh của Generative AI (Nah và cộng sự, 2023) trong phân tích ngữ nghĩa sâu và kỹ thuật Prompt Engineering nâng cao để tự động hóa quy trình lọc tin. Cấu trúc bài báo được tổ chức chặt chẽ nhằm làm sáng tỏ các luận điểm nghiên cứu. Bối cảnh bài toán về sự bùng nổ thông tin và hạn chế của các phương pháp lọc hiện tại được trình bày trong phần Giới thiệu. Trong Mục 2, chúng tôi trình bày chi tiết kiến trúc hệ thống phân lớp đề xuất, bao gồm các mô hình toán học định lượng cho hàm quyết định lọc tin và giải thuật theo dõi hành vi người dùng . Mục 3 tập trung vào quá trình thiết kế, hiện thực hóa giải pháp công nghệ trên nền tảng di động và phân tích hiệu quả thực nghiệm thông qua các chỉ số định lượng cụ thể. Cuối cùng, các đóng góp chính của đề tài cùng định hướng tối ưu hóa kỹ thuật trong tương lai được trình bày trong Mục 4.

2. MÔ HÌNH VÀ PHƯƠNG PHÁP NGHIÊN CỨU

Nghiên cứu này đề xuất một kiến trúc hệ thống tích hợp (integrated system architecture) kết hợp giữa sức mạnh tính toán, lưu trữ đám mây và khả năng suy luận của các mô hình ngôn ngữ lớn. Phương pháp nghiên cứu tập trung vào việc giải quyết bài toán cốt lõi: Tự động hóa quy trình kiểm duyệt nội dung tin tức đa chiều bằng AI.



Hình 1. Kiến trúc tổng quan hệ thống Safe News

2.1. Kiến trúc hệ thống và Công nghệ lõi

Kiến trúc được xây dựng dựa trên mô hình Client-Server hiện đại, bao gồm ba phân hệ chính như Hình 1, hoạt động không đồng bộ nhưng liên kết chặt chẽ về dữ liệu:

Phân hệ Thu thập và tiền xử lý dữ liệu: được thiết kế nhằm mục đích cô lập sự biến động của nguồn dữ liệu bên ngoài khỏi tính ổn định của hệ thống lõi. Trong kiến trúc này, các mô-đun Crawler thực hiện cơ chế trích xuất dữ liệu không đồng bộ từ đa nguồn, sau đó chuẩn hóa các định dạng HTML không đồng nhất thành cấu trúc JSON tiêu chuẩn. Nhóm nghiên cứu quyết định tách biệt hoàn toàn tầng này thay vì tích hợp trực tiếp vào Backend server để đảm bảo nguyên tắc "Cô lập lỗi": nếu cấu trúc một trang báo nguồn thay đổi hoặc gặp sự cố, thì sẽ không gây ra hiệu ứng làm sập toàn bộ hệ thống phục vụ người dùng, đồng thời cho phép bảo trì nóng các trình thu thập mà không cần gián đoạn dịch vụ.

Phân hệ xử lý lọc nội dung và xử lý bằng AI: đây là phân hệ điều phối trung tâm, nơi tích hợp logic nghiệp vụ với các microservices xử lý trí tuệ nhân tạo để đảm bảo chất lượng dữ liệu đầu ra. Chúng tôi lựa chọn kiến trúc Server-side Processing cho mô-đun AI thay vì On-device Processing nhằm tối ưu hóa hiệu năng thiết bị máy khách và đảm bảo tính nhất quán của thuật toán; mọi cập nhật về mô hình AI đều được triển khai trong suốt tại server mà không yêu cầu người dùng phải cập nhật phiên bản ứng dụng mới.

Chiến lược lưu trữ được xây dựng dựa trên Firebase để lưu trữ dữ liệu dưới dạng document JSON, cho phép hệ thống thích ứng nhanh chóng với sự thay đổi về thuộc tính của các bài báo (ví dụ: thêm video, tags mới) mà không cần thực hiện các thao tác di chuyển dữ liệu phức tạp như trong RDBMS truyền thống. Quyết định thiết kế này được đưa ra dựa trên phân tích đặc thù dữ liệu tin tức: dữ liệu có tính thời gian thực cao, cấu trúc đa dạng và yêu cầu khả năng mở rộng chiều ngang để đáp ứng lượng dữ liệu tích lũy lớn theo thời gian.

Phân hệ giao diện người dùng: được kiến trúc theo mô hình Thin-client, tuân thủ nghiêm ngặt nguyên tắc "Phân tách mối quan tâm" bằng cách loại bỏ hoàn toàn logic xử lý phức tạp khỏi giao diện người dùng. Được xây dựng trên nền tảng Flutter, phân hệ này chỉ đóng vai trò hiển thị và tương tác, giao tiếp với Backend thông qua chuẩn RESTful API. Việc giảm tải tính toán cho máy khách là quyết định then chốt giúp ứng dụng đạt được độ trễ thấp và trải nghiệm mượt mà trên các thiết bị di động cấu hình thấp, đồng thời đơn giản hóa quy trình phát triển đa nền tảng (Android/iOS).

2.2. Mô hình hóa toán học và các giải thuật tương ứng

2.2.1. Công thức quyết định lọc tin tức:

Gọi x là một bài báo đầu vào. Kết quả phân tích từ GenAI trả về vectơ đặc trưng $V(x) = (s, t, w)$, trong đó:

- $s \in \{-1, 0, 1\}$: Điểm cảm xúc, tương ứng với {Tiêu cực, Trung tính, Tích cực}.
- $t \in \{0, 1\}$: Nhãn độc hại (Is_Toxic), với 1 là Độc hại, 0 là An toàn.
- $w \in \{0, 1\}$: Nhãn cảnh báo (Is_Warning), với 1 là tin tiêu cực nhưng có giá trị cảnh báo (ví dụ: bão lũ, dịch bệnh), 0 là không có.

Dựa trên vectơ đặc trưng ngữ nghĩa $V(x) = (s, t, w)$, Hàm quyết định $F(x)$được thiết lập như sau:

$$F(x) = \begin{cases} 1, & if\ t = 0 \wedge (s \geq 0 \vee (s = -1 \wedge w = 1)) \\ 0, & if\ t = 1 \vee (s = -1 \wedge w = 0) \end{cases}$$

Công thức $F(x)$ được thiết kế chặt chẽ nhằm giải quyết bài toán cốt lõi của việc lọc tin:

Điều kiện Loại bỏ ($F(x) = 0$): Hệ thống sẽ ngay lập tức từ chối bài viết nếu bản tin chứa nội dung độc hại, vi phạm tiêu chuẩn ($t = 1$) hoặc đó là một bản tin tiêu cực mang tính chất giật gân, không mang lại giá trị cảnh báo ($s = -1 \wedge w = 0$). Đây chính là nguyên nhân cốt lõi gây ra hiệu ứng Doomscrolling.

Điều kiện Chấp nhận ($F(x) = 1$): Bài viết được lưu lại khi thỏa mãn tiêu chí an toàn cơ sở ($t = 0$), đồng thời cảm xúc bài viết ở mức trung tính hoặc tích cực ($s \geq 0$). Điểm sáng của phương trình nằm ở việc xử lý trường hợp ngoại lệ: Nếu bài viết mang tính tiêu cực ($s = -1$) nhưng được GenAI dán nhãn là thông tin cảnh báo quan trọng (như bão lũ, dịch bệnh với $w = 1$), hệ thống vẫn chấp nhận ($F(x) = 1$). Điều này đảm bảo hệ thống không "lọc nhầm" các tin tức thiết yếu của xã hội.

2.2.2. Giải thuật Thu thập và Phân tích Tin tức

Quy trình xử lý tin tức được tự động hóa hoàn toàn thông qua thuật toán được mô tả dưới đây.

Thuật toán 1: Quy trình Thu thập và Lọc Tin tức.

Input: Danh sách RSS URLs (L_RSS), Cache (C)
Output: Cập nhật Firestore Database (DB)
1. **Foreach** url **in** L_RSS do
2. articles ← FetchRSS(url)
3. **Foreach** item in articles do
4. hash ← MD5(item.link)
5. **If** DB.Exists(hash) **then** Continue
6. content ← ExtractContent(item.link)
7. *// Gọi Gemini API để phân tích*

```
8.        result ← Gemini.Analyze(content, prompt_template)
9.        // Áp dụng hàm lọc F(x)
10.       If result.is_toxic == 1 OR (result.sentiment == -1 AND result.is_warning == 0) then
11.           Log("Rejected: " + item.title)
12.       Else
13.       news_obj ← CreateObject(item, result.summary,
                result.sentiment)
14.           DB.Save(news_obj)
15.       End If
16.     End Foreach
17.   End Foreach
```

Thuật toán được thiết kế để giải quyết vấn đề phân loại đối với luồng dữ liệu tin tức đầu vào không cấu trúc. Mục tiêu là xác định nhãn $L \in \{\text{Safe, Unsafe}\}$ cho mỗi bài báo N dựa trên nội dung văn bản và ngữ cảnh, sử dụng AI làm công cụ suy luận. Thuật toán vận hành dựa trên cơ chế suy luận ngữ nghĩa thay vì đối sánh từ khóa truyền thống. Quy trình xử lý được chia thành ba giai đoạn trọng yếu:

Dữ liệu đầu vào từ các nguồn không đồng nhất thường chứa các nhiễu như thẻ HTML dư thừa, quảng cáo hoặc mã script. Bước tiền xử lý có nhiệm vụ chuẩn hóa văn bản về dạng plain text để tối ưu hóa token đầu vào cho mô hình AI, giảm thiểu chi phí tính toán và tăng độ chính xác của ngữ cảnh.

Đây là trọng tâm của thuật toán. Thay vì sử dụng các mô hình phân loại truyền thống (như SVM hay Naive Bayes), thuật toán tận dụng khả năng hiểu ngôn ngữ tự nhiên của AI tạo sinh (Gemini). Tham số P (Prompt) đóng vai trò như một hàm truyền, định hướng mô hình không chỉ phân loại mà còn thực hiện tóm tắt và trích xuất thực thể trong cùng một lần gọi. Yêu cầu đầu ra bắt buộc là định dạng JSON để đảm bảo tính cấu trúc khi tích hợp vào hệ thống Backend.

Cơ chế kiểm soát được áp dụng tại bước cuối cùng. Chỉ những bản ghi có nhãn True và độ tin cậy vượt ngưỡng quy định mới được chuyển tiếp vào cơ sở dữ liệu. Cơ chế này loại bỏ triệt để các tin tức mang tính kích động, tin giả hoặc nội dung gây hiệu ứng Doomscrolling, đảm bảo tính toàn vẹn của hệ sinh thái tin tức "sạch".

2.2.3. Công thức truy hồi tính Streak:

Giá trị chuỗi (Streak) của ngày d được tính dựa trên trạng thái đọc của ngày hôm đó và giá trị Streak của ngày hôm trước $(d-1)$

$$S_d = \begin{cases} S_{d-1} + 1, & if\ C_1\ (Duy\ trì) \\ S_{d-1}, & if\ C_2\ (Chưa\ xong) \\ 1, & if\ C_3\ (Bắt\ đầu) \\ 0, & otherwise\ (Mất\ chuỗi) \end{cases}$$

Trong đó:

- $C_1: R_d = 1 \wedge R_{d-1} = 1$
- $C_2: R_d = 0 \wedge t < t_{end}$
- $C_3: R_d = 1 \wedge R_{d-1} = 0$
- $R_d \in \{0,1\}$: Trạng thái đọc báo của người dùng vào ngày d.
- $R_d = 1$ nếu tổng thời gian đọc trong ngày $\sum T_{read} \geq 30$ giây.

Tính toán chuỗi phụ thuộc vào trạng thái R_d, trong đó $R_d = 1$ chỉ khi người dùng có thời gian đọc $T_{read} \geq 30$ nhằm ngăn chặn hành vi "lướt qua" đối phó.

$C_1(R_d = 1 \wedge R_{d-1} = 1)$: Người dùng hoàn thành mục tiêu đọc trong cả ngày hôm nay và hôm qua, chuỗi được cộng dồn. $C_2(R_d = 0 \wedge t < t_{end})$: Trong ngày hiện tại (thời gian t chưa qua ngày mới t_{end}), người dùng chưa đạt đủ 30 giây đọc. Hệ thống tạm giữ nguyên chuỗi cũ S_{d-1} để chờ người dùng hoàn thành.

$C_3(R_d = 1 \wedge R_{d-1} = 0)$: Người dùng đạt mục tiêu hôm nay nhưng đã bỏ lỡ ngày hôm qua. Chuỗi cũ bị gãy và bắt đầu chu kỳ mới với S_{d-1}.

2.2.4. Giải thuật Gamification và Theo dõi Tương tác

Thuật toán 2: Theo dõi Phiên đọc và Cập nhật Streak

Input: User (U), Article (A), Timer (T)
Output: Cập nhật User Stats
1. **StartTimer**(T)
2. **While** UserIsReading(A) **do**
3. **If** T.elapsed ≥ 30s **then**
4. valid_read ← True
5. Break
6. **End If**
7. **End While**
8. **If** valid_read **then**
9. last_read ← U.last_read_date
10. **If** IsYesterday(last_read) **then**
11. U.streak ← U.streak + 1
12. **Else** If IsToday(last_read) **then**
13. // Do nothing
14. **Else**
15. U.streak ← 1
16. **End If**
17. U.total_read ← U.total_read + 1
18. **DB**.Update(U)
19. **End If**

Thuật toán được thiết kế để định lượng mức độ tương tác của người dùng với nội dung "sạch". Mục tiêu là khuyến khích thói quen đọc tin tích cực mỗi ngày thông qua cơ chế khen thưởng tâm lý. Bài toán đặt ra là xác định chính xác một "phiên đọc hợp lệ" dựa trên tham số thời gian thực và cập nhật trạng thái chuỗi (Streak) của người dùng theo logic truy hồi.

Thuật toán sử dụng một bộ đếm thời gian (T) chạy ngầm. Điều kiện $T.elapsed \geq 30s$ là bộ lọc quan trọng để phân biệt giữa hành vi "lướt qua" và "đọc hiểu". Điều này đảm bảo chỉ số Streak phản ánh đúng chất lượng tiếp nhận thông tin của người dùng.

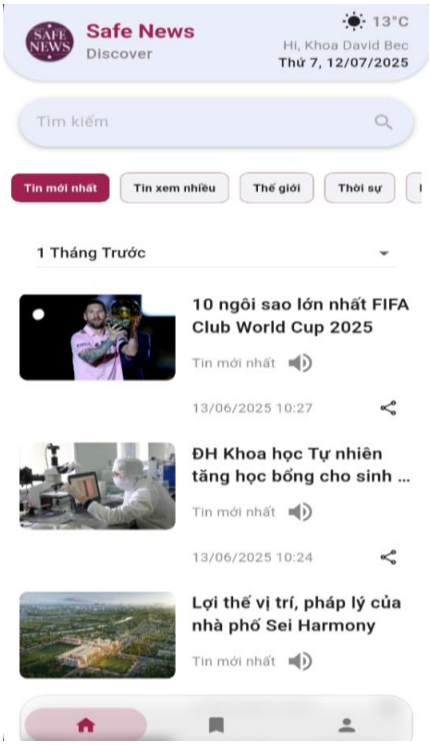
Trong kiến trúc MVVM (Epiloksa và cộng sự, 2022) của ứng dụng Flutter, thuật toán này được đặt tại tầng ViewModel. Nó lắng nghe trạng thái cuộn trang và thời gian hiển thị của View, sau đó tính toán logic và chỉ gửi lệnh cập nhật (DB.Update) xuống tầng Model khi và chỉ khi điều kiện valid_read được thỏa mãn. Cách thiết kế này giúp giảm thiểu số lượng Write Request không cần thiết xuống Firebase, tối ưu hóa chi phí băng thông và năng lượng thiết bị.

3. KẾT QUẢ NGHIÊN CỨU

3.1. Thiết kế và hiện thực hệ thống

Safe News được thiết kế theo kiến trúc nhiều lớp, kết hợp giữa backend thu thập dữ liệu, thành phần phân tích nội dung sử dụng mô hình sinh ngôn ngữ và ứng dụng khách trên di động. Tầng backend chịu trách nhiệm thu thập bài báo từ nhiều nguồn RSS khác nhau, chuẩn tích và lưu trữ kết quả vào cơ sở dữ liệu đám mây. Thiết kế này tách biệt rõ ràng các mối quan tâm: việc crawl dữ liệu, suy luận nội dung và cung cấp dịch vụ cho client được đóng gói trong các mô-đun riêng, dễ mở rộng và bảo trì.

Ứng dụng di động được xây dựng bằng Flutter và áp dụng kiến trúc MVVM để tách biệt phần giao diện, logic trình bày và dữ liệu miền như Hình 1 và giao diện thử nghiệm thực tế như Hình 2. Cách tổ chức này cho phép tái sử dụng logic trên nhiều màn hình, hỗ trợ kiểm thử đơn vị cho ViewModel, đồng thời thuận tiện khi mở rộng thêm chức năng (ví dụ gợi ý bài viết, thống kê người dùng) mà không làm vỡ cấu trúc tổng thể.



Hình 2. Giao diện ứng dụng hiện thực từ kiến trúc

3.2. Quy trình lọc nội dung sử dụng AI Tạo sinh

Quy trình lọc nội dung được hiện thực dưới dạng pipeline gồm các bước: trích xuất nội dung (tiêu đề, mô tả, thân bài), xây dựng prompt theo mẫu cố định, gửi yêu cầu phân tích đến mô hình GenAI và cuối cùng là ánh xạ kết quả trả về thành các nhãn và điểm số phục vụ quyết định lọc. Mô-đun này được đóng gói như một dịch vụ độc lập, giúp dễ dàng thay thế mô hình hoặc điều chỉnh cấu hình mà không ảnh hưởng đến các phần còn lại của hệ thống.

Hệ thống trích xuất nội dung định kỳ lấy dữ liệu RSS, chuẩn hóa thành "News Object", sau đó gửi tiêu đề và URL sang mô hình Gemini để phân tích nội dung, đánh giá cảm xúc và đánh dấu tính "toxic". Kết quả phân tích được minh họa như Bảng 1 và được lọc để lưu trữ trong CSDL.

Bảng 1. Minh họa kết quả phân tích ngữ nghĩa và quyết định lọc tin thực tế

STT	Tiêu đề rút gọn	Mô tả ngắn	Kết quả
1	"Kỷ lục gần 23.000 thí sinh tranh suất vào trường Công an"	Tin giáo dục, kỳ thi, thông tin tích cực về chọn ngành nghề	Lưu
2	"Điều chỉnh giá tính lệ phí trước bạ cho gần 450 ôtô, xe máy"	Thông tin chính sách tài chính, trung tính	Lưu
3	"1.000 quả pháo hoa thắp sáng bầu trời Hạ Long"	Tin thời sự – văn hóa, không tiêu cực	Lưu
4	"Chủ quán game hành hung hai thiếu niên bằng nhiều vật dụng"	Bạo lực, mô tả hành vi gây hại, ảnh hưởng tâm lý	Không
...	.....	.....	...

Dữ liệu minh chứng cho thế mạnh của mô hình AI tạo sinh so với các phương pháp lọc từ khóa truyền thống. Kết quả phân tích từng trường hợp cụ thể cho thấy hệ thống không chỉ dừng lại ở việc nhận diện bề mặt văn bản mà đã đạt được khả năng "thấu hiểu" ngữ cảnh. Điển hình là khả năng phân biệt tinh tế giữa các bản tin chứa từ khóa nhạy cảm nhưng mang ý nghĩa tích cực (như tin về pháo hoa) và các tin tức thực sự độc hại (như bạo lực). Quyết định "Lưu" hay "Loại

bỏ" ở đây không còn là một phép so sánh chuỗi (string matching) cứng nhắc, mà là kết quả của một quá trình suy luận logic (logical inference), đảm bảo loại bỏ triệt để các nội dung vi phạm tiêu chuẩn cộng đồng mà không làm nghèo đi sự đa dạng của dòng tin tức.

Chúng tôi xem xét bài báo mới mỗi ngày với số lượng 30 bài ngẫu nhiên như trong Bảng 2. Hệ thống sử dụng một mô đun độc lập để thu thập dữ liệu thời gian thực từ các luồng RSS và API công khai của các trang báo điện tử chính thống. Quy trình trích xuất bao gồm việc bóc tách tiêu đề, đoạn trích dẫn và nội dung cốt lõi, đồng thời loại bỏ các thẻ HTML nhiễu.

Trong giai đoạn đánh giá thực nghiệm, lưu lượng thu thập được cấu hình giới hạn ở mức 30 bài báo mỗi ngày. Ngưỡng thiết lập này không phải là giới hạn năng lực của mô đun thu thập dữ liệu, mà là một quyết định thiết kế có chủ đích dựa trên ba cơ sở khoa học sau:

Việc gọi trực tiếp API của mô hình ngôn ngữ lớn cho mỗi bản tin phát sinh chi phí tính toán và chịu ràng buộc về giới hạn truy vấn. Mức 30 bài/ngày là quy mô tối ưu cho môi trường thử nghiệm nhằm đánh giá tính ổn định mà không gây tắc nghẽn API.

Dưới góc độ trải nghiệm, 30 bản tin mới mỗi ngày là khối lượng thông tin hoàn toàn đủ sức lấp đầy giao diện cuộn của người dùng trong một phiên đọc tiêu chuẩn. Mẫu dữ liệu này đảm bảo bao phủ đủ các kịch bản kích hoạt hàm quyết định $F(x)$, phục vụ mục tiêu cốt lõi là ngăn chặn hành vi doomscrolling.

Kết quả phân loại trong mỗi ngày cho thấy kết quả chi tiết như sau:

Bảng 2. Thống kê hiệu năng phân loại và tỷ lệ lọc trên tập dữ liệu thử nghiệm

Tiêu chí đánh giá	Số lượng bài báo	Tỷ lệ (%)	Phân tích ý nghĩa
Tổng bài thu thập	30	100%	Tập dữ liệu đầu vào đa dạng chủ đề (Xã hội, Thể thao, Kinh doanh, Pháp luật).
Số bài được chấp nhận	24	80%	An toàn hoặc Tích cực.
Số bài loại bỏ	6	20%	Các bài báo cảnh báo.

Độ chính xác trên tập tin: Trong số 24 bài được hệ thống lưu lại, kiểm tra thủ công cho thấy khoảng 80% là thực sự phù hợp với tiêu chí lọc tin. Tuy nhiên, vẫn còn khoảng 20% (tương đương 4-5 bài) là các tin tức vẫn mang sắc thái tiêu cực nhẹ hoặc gây tranh cãi mà AI chưa lọc triệt để.

Ngoài độ chính xác về mặt nội dung, hiệu năng kỹ thuật cũng là yếu tố then chốt cho trải nghiệm người dùng.

Thời gian xử lý trung bình: Thời gian để Gemini phân tích và trả về kết quả cho một bài báo dao động từ 2.1s đến 2.5s. Với tần suất thu thập tin 30 phút/lần, tốc độ này hoàn toàn đáp ứng được nhu cầu cập nhật tin tức gần như thời gian thực mà không gây tắc nghẽn hệ thống.

Hiệu quả của cơ chế Cache: Việc sử dụng mã băm MD5 để kiểm tra trùng lặp trước khi gọi API AI đã giúp giảm khoảng 40-50% số lượng request không cần thiết đến Gemini, giúp tiết kiệm chi phí vận hành đáng kể.

Hiệu năng ứng dụng Flutter: Ứng dụng client đạt tốc độ khung hình ổn định 60fps trên các thiết bị thử nghiệm (Android/iOS). Kiến trúc MVVM giúp tách biệt luồng dữ liệu, đảm bảo giao diện không bị dừng khi đang tải dữ liệu nặng hoặc xử lý logic ngầm.

4. KẾT LUẬN

Nghiên cứu này đã hoàn thiện việc thiết kế và triển khai thực tế hệ thống "Safe News", minh chứng cho tính khả thi của việc tích hợp các mô hình toán học định lượng với năng lực suy luận của AI tạo sinh trong bài toán xử lý thông tin tự động. Các đóng góp khoa học và công nghệ chính của đề tài bao gồm: (i) hiện thực hóa thành công hàm quyết định lọc $F(x)$. Đây là thành phần lõi giúp tự động hóa hoàn toàn quy trình kiểm duyệt nội dung, đạt độ chính xác thực nghiệm 80% trên tập dữ liệu kiểm thử, khẳng định sự ưu việt so với các phương pháp lọc từ khóa truyền

thống; (ii) tích hợp sâu cơ chế Gamification dựa trên thuật toán phân tích hành vi người dùng theo thời gian thực. Tính năng này đóng vai trò can thiệp kỹ thuật trực tiếp, góp phần giảm thiểu hiệu ứng tâm lý tiêu cực từ hành vi Doomscrolling.

Hiện tại nghiên cứu này đang tập trung chủ yếu vào việc đề xuất "kiến trúc hệ thống tích hợp GenAI" và kiểm chứng giải thuật Gamification. Hạn chế của mô hình thử nghiệm hiện tại là chưa tiến hành đo lường các chỉ số phân loại chuyên sâu của AI trên một tập dữ liệu quy mô lớn, cũng như chưa thực hiện đối sánh toàn diện với các mô hình học máy truyền thống. *Trong tương lai:* đây sẽ là trọng tâm trong định hướng phát triển tương lai của nhóm nghiên cứu, nhằm tối ưu hóa tỷ lệ phân loại sai củng cố độ tin cậy của hàm quyết định trước khi triển khai thực tế. Đồng thời, kiến trúc hệ thống sẽ được nâng cấp để hỗ trợ tích hợp đa nguồn dữ liệu quốc tế, hướng tới việc đa dạng hóa hệ sinh thái nội dung số.

TÀI LIỆU THAM KHẢO

Shabahang, R., Saeb, A. M., Becker, M., & Reyes, M. E. S. (2021). Online news addiction: Future anxiety, fear of missing out on news, and interpersonal trust contribute to excessive online news consumption. Online Journal of Communication and Media Technologies, 11(2), Article e202105.

Nguyễn, T. G. (2020). Cảm giác lo âu và truyền thông đương đại: Một tổng quan điểm luận [Anxiety and contemporary media: A literature review]. Vietnam Journal of Social Sciences and Humanities, 6(3), 324–336.

Dixit, K., & Ashutosh, D. K. (2025). Collective stress during crisis-based doomscrolling. TPM–Testing, Psychometrics, Methodology in Applied Psychology, 32(S7), 2751-2763.

Fu, J., Qu, Y., & Wang, Z. (2009). Two level question classification based on SVM and question semantic similarity [Paper presentation]. 2009 International Conference on Electronic Computer Technology, Macau, China. IEEE.

Johnsen, M. (2024). Large language models (LLMs). Maria Johnsen.

Gabriel, S., Lyu, L., Siderius, J., Ghassemi, M., Andreas, J., & Ozdaglar, A. (2024). MisinfoEval: Generative AI in the era of "alternative facts". arXiv. https://doi.org/10.48550/arXiv.2410.09949

Sandrini, L., & Somogyi, R. (2023). Generative AI and deceptive news consumption. Economics Letters, 232, Article 111317.

Campante, F. R., et al. (2024). GenAI Misinformation, Trust, and News Consumption: Evidence from a Field Experiment. NBER Working Paper Series, w34100.

Chang, Y., & Wang, X. (2025). The Ethical Risks of Generative AI News from the Perspective of Human-Machine Collaborative Narration. *Atlantis Press*.

Loth, A., Kappes, M., & Pahl, M. O. (2024). Blessing or curse? A survey on the impact of generative AI on fake news. arXiv. https://doi.org/10.48550/arXiv.2404.03021

Kieslich, K., Diakopoulos, N., & Helberger, N. (2025). Anticipating impacts: Using large-scale scenario-writing to explore diverse implications of generative AI in the news environment. AI and Ethics, 5(5), 4555–4577.

Nah, F. F.-H., Zheng, R., Cai, J., Siau, K., & Chen, L. (2023). Generative AI and ChatGPT: Applications, challenges, and AI-human collaboration. Journal of Information Technology Case and Application Research, 25(3), 277–304.

Epiloksa, H. A., Kusumo, D. S., & Adrian, M. (2022). Effect of MVVM architecture pattern on Android based application performance. Jurnal Media Informatika Budidarma, 6(4), 1949–1955.