

CHAPTER 4 – RESULT AND DISCUSSION

4.1. Introduction

This chapter will describe the results of the methods we propose on Chapter 3 as previously mentioned.

4.2. Preparation Phase

This section acts as the initial foundation of the research, with the main aim to set specific backgrounds and operational configurations of the CMS system which is the object of study. This process includes determining the research context by formulating the scope of the assessment, identifying the main stakeholders, and conducting a review of relevant documentation. These steps are important to ensure that research has a strong and structured foundation.

The expected results of this phase are clearly defined context documents, which include the description of the operational environment, the use of the system, and CMS data needed to conduct a comprehensive assessment. This document will be the main reference in the stages of further research, ensuring that every aspect of the assessment is carried out with a deep understanding of the system and the environment.

4.2.1 Define Objective

This research wants to identify potential threats to the CMS application and provide appropriate countermeasures. Following this, the research will highlight the gaps in the current system, providing insights to assist organizations in achieving base on secure control with OWASP Top 10 and ISO 27001:2022.

This phase focused on establishing the context for the assessment. This involved a detailed analysis of the application's environment, including its functional requirements and compliance objectives goal.

4.2.1.1. Business Security Goals

This phase focused on establishing the context for the a business security goals. This involved a detailed analysis of the application's environment, including its functional requirements and compliance objectives goal.

Based on information gathered from discussion stakeholder and legacy documents, PT XYZ relies heavily on digital newsroom and Content Management System (CMS), business security goals are a set of strategic targets designed to create strong and reliable application security in supporting the company's business.

Key finding business security goals:

1. Maintain core business continuity,
2. Protect critical information values,
3. Prevent reputational and financial losses due to cyber incidents.

These goals are in line with the basic principles of information security, namely Confidentiality, Integrity, and Availability (CIA). However, in the context of PT XYZ's business organization, **the goals are expanded to include journalism interests, content legality, and distribution platform resilience.**

The following are the results of interviews regarding business security goals for CMS journalists, which can be summarized in the table 4.1:

Table 4.1. Business Security Goals for CMS journalist.

Security Goal	Description	Context of CMS
1. Availability	Ensuring the CMS system is always active and accessible to journalists, editors, and production teams at any time, especially during breaking news conditions.	CMS downtime can result in news not being broadcast on time and disrupt the live broadcast cycle.
2. Integrity	Ensure that news content created, edited and published is not changed without authorization or manipulated by unauthorized parties.	Unauthorized modified articles or videos may lead to disinformation or defamation.
3. Confidentiality	Protect sensitive data such as sources, investigative documents, and user credentials from being leaked to the public or competitors.	CMS user access should be strictly regulated according to role: journalist, editor, admin.

Security Goal	Description	Context of CMS
4. Auditability & Accountability	Able to track every user activity on the system to ensure responsibility, especially in the event of an incident or error in publication.	Logging user activity and revised versions of articles is important for forensic investigations.
5. Resilience	Ensures the CMS system continues to run even in the event of an attack or technical disruption through backup, failover, and disaster recovery plans.	Have and enable backup servers and automatic recovery in case of ransomware or hardware failure.
6. Compliance	Ensure CMS complies with national and international security regulations such as ISO27001, ITE Law, PDP, and broadcasting regulations.	User data storage, access logs, and publication controls comply with legal and ethical policies for digital journalism and applications comply with cybersecurity policies such as OWASP Top 10 and ISO 27001:2022.
7.Brand and Reputational Protection	Ensuring the CMS system runs efficiently, does not complicate the work of journalists, and continues to function under all operational conditions.	A secure but slow CMS will hamper the speed of news production, so a balance is needed between security and usability.

4.2.1.2. Risk Criteria Stakeholder

Based on table 4.1 Business Security Goals for CMS journalist, the current security conditions are as follows:

Table 4.2. Condition Category Current Security of CMS

Condition Category	Description
Availability	Based on reports the complex of the system has experienced down and cannot be accessed by the user. Operations are disturbed. Appendix 13
Integrity	The existence of important assets such as articles, videos, images that are lost due to modifications to the content due to the weak application system.
Confidentiality	Some assets, such as videos or images, were misused by some users without editorial permission. There have been instances where videos disappeared from CMS applications.

Based on Table 4.2, the security conditions and fulfillment of expectations of the PT XYZ company's goals business goals, a method is needed in increasing or minimizing the risk by applying risk management to the CMS application.

Risk management of Content Management System (CMS) applications used in journalist workflows in television media companies requires comprehensive risk assessment criteria to identify and classify the severity and urgency of each potential threat. These risk criteria are compiled based on a combination of technical and business aspects, taking into account the typical characteristics of the media industry that emphasizes the speed of information distribution, content accuracy, and public trust.

The first criterion is the impact on operations, namely the extent to which disruption to the CMS affects the continuity of news production and publication. A disrupted system can cause delays in news broadcasts, loss of information momentum, and failure of cross-platform distribution. Furthermore, there is an impact on reputation, which includes the potential for damage to the institution's image due to publication errors, content manipulation, or internal information leaks. Reputation is a critical asset for television media that relies on public trust and editorial integrity.

The next criterion is the threat exploitability criterion, which assesses how easily a risk can be exploited by threat actors, either through technical loopholes or user negligence. Therefore, risk sources are classified into internal (such as user error, insider threats) and external (such as attacks from third parties), each of which requires a different mitigation approach. In addition, compliance with regulations such as UU PDP, ISO 27001, and the ITE Law is also an important element in risk assessment, considering that CMS manages user data, news archives, and digital activity logs that are vulnerable to misuse if not managed properly.

After knowing the criteria above, the next step is to collect data on the application system architecture or infrastructure to conduct data flow analysis which will be used to determine threats and vulnerabilities in the system.

4.3. Data Collection.

Based on information gathered from interviews and documents manual CMS, the application has several features regarding add input news, news content, projection and report news. This research will focus on two distinct workflows: one designed for Operations journalist news input and the other designed for stream news to production. See figure 4.1 for the architecture diagram.

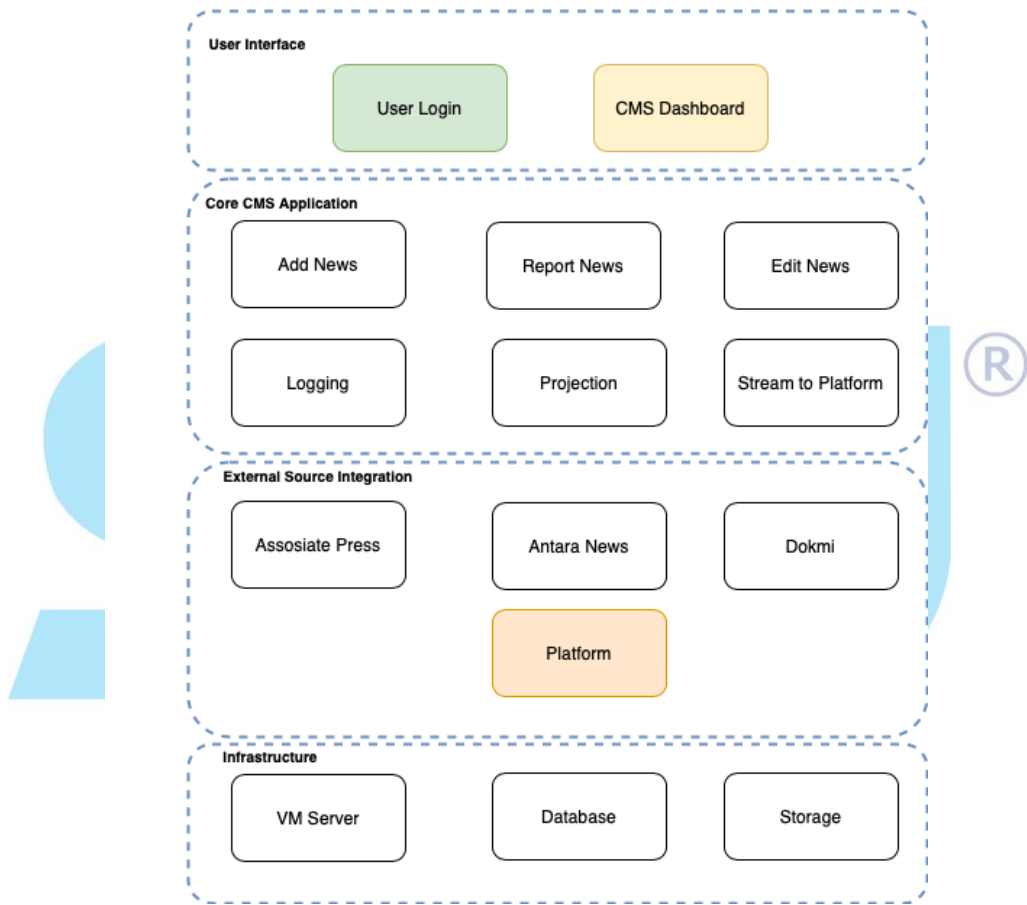


Figure 4.1. Architecture Diagram of CMS

The diagram presents at Figure 4.1 is a structured layout of a Content Management System (CMS), segmented into various operational layers, each designated for specialized tasks within the CMS framework.

At the topmost level, the User Interface layer encompasses the Login Page, facilitating secure user authentication, and the CMS Dashboard, which serves as the central hub for monitoring and managing journalist interactions. Below this, the External Integrations

layer highlights the system’s connectivity with third-party services for resource like Asosiate Press, Dokmi or Antara Press.

4.3.1.Decompose Application

The decompose application process in threat modeling is a fundamental stage that aims to break down the system into its components in order to comprehensively understand the architecture, data flow, and points of interaction that have the potential to be targets of attacks. This step begins by identifying the main components of the system, including core features such as login, content management, and media upload, as well as supporting technologies such as programming languages, frameworks, and databases used.

Next, data flow mapping is carried out using Data Flow Diagrams (DFD) to visualize the movement of information from user input to data storage and processing, including identifying entry points and exit points that are the entrances and exits of data. In addition, determining trust boundaries is crucial to mark areas where the level of trust changes, such as in the process of validating user input or interacting with third-party APIs. The analysis also includes mapping access rights (authorization boundaries) to ensure the principle of least privilege is applied, as well as documenting critical assets such as sensitive data or system configurations that are the main targets of attackers.

4.3.2.Identification Component and Critical Asset

The CMS application encompasses a variety of assets that are integral to its functionality and security. In this assessment, researchre specifically focus on the following key assets on Table 4.2. Each of these assets plays a crucial role in the application’s overall performance and security posture, and their thorough evaluation is essential for understanding the application’s vulnerability to potential cybersecurity threats.

There are Component and Asset CMS PT XYZ :

Table 4.3. Component CMS

No	Component	Description	Exposure	Connection
1.	User Browser	Users access the cms application through their web browser via https.	Public	Sends/receives data to/from application.

No	Component	Description	Exposure	Connection
2.	Web Login	a) Serves as the interface for users. b) Sends auth login to the backend services. c) Communicates with the backend to other services.	Public	Sends/receives data to/from database and API Gateway.
3.	API Gateway	d) Generates and verifies authentication tokens to database and storage. e) Handles business logic, including authentication, transaction processing, and data storage.	Internal	Communicates directly with the central database ,possibly consumed data storage, external source and stream to production system.
4.	Database	f) Generates and verifies authentication tokens to database and data storage. g) Handles business logic, including authentication and metadata news content.	Internal	API Gateway Communicates directly with the central database and storage.
5.	Storage	h) Logging store i) Place of data photos, videos j) Storage communication to system base on samba share.	Internal	API Gateway Communicates directly with the central database and Possibly consumed.
6.	External Source	k) Place of Articles from external resource like Antara, AP and Dokmi. l) Photos and content from AP,ANTARA.	External	Communicates directly with the API Gateway.
7.	Publish Content to Production.	m) Generates and verifies authentication tokens publish to production.	Internal	Communicates directly with the API Gateway to other system.

4.3.3. Data Flow Diagram

Based on the CMS components and assets in **Table 4.3**, the data flow diagram can be described. The following is the Data Flow Diagram for system level 0.

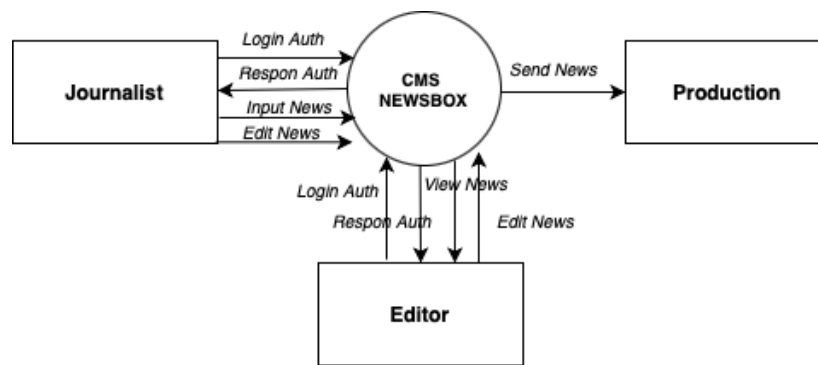


Figure 4.2. Data Flow Diagram Level 0 CMS

From the explanation of the main process above, the data flow diagram can be described as follows. This is the process of using the application with a focus on **news input** and **news streaming to the platform** as follows:

A. News Input Process.

The first time **the journalist** logs in to the application then after successfully logging in the journalist fills in the coverage results by accessing the add coverage feature. in the coverage form, the journalist must fill in the crew if any, desk type, title, and news content, can be added shotlist, add video link if any and photo attachments if any. finally submit to the system. the results of the coverage can be seen in the coverage results feature.

B. Stream News to Platform .

The **editors** log in to the application then after successfully logging in, several news coverage results can be seen in the coverage results menu. The editors can select news by clicking on one of the news titles on the list. Then the editors can edit the news, the suitability of the video or photo if it is correct, the editors can send it to the desired platform such as website, television, epaper, social media by clicking the submit button on each platform. Then the usage record will be displayed as a usage log.

In **Figure 4.2** above, there are 2 entities that play a role, namely **journalists** inputting news into the CMS and **editors** managing news and selecting news to be distributed to production. In accordance with the process above, the data flow diagram can be described as follows:

4.3.3.1.Data Flow News Input Process.

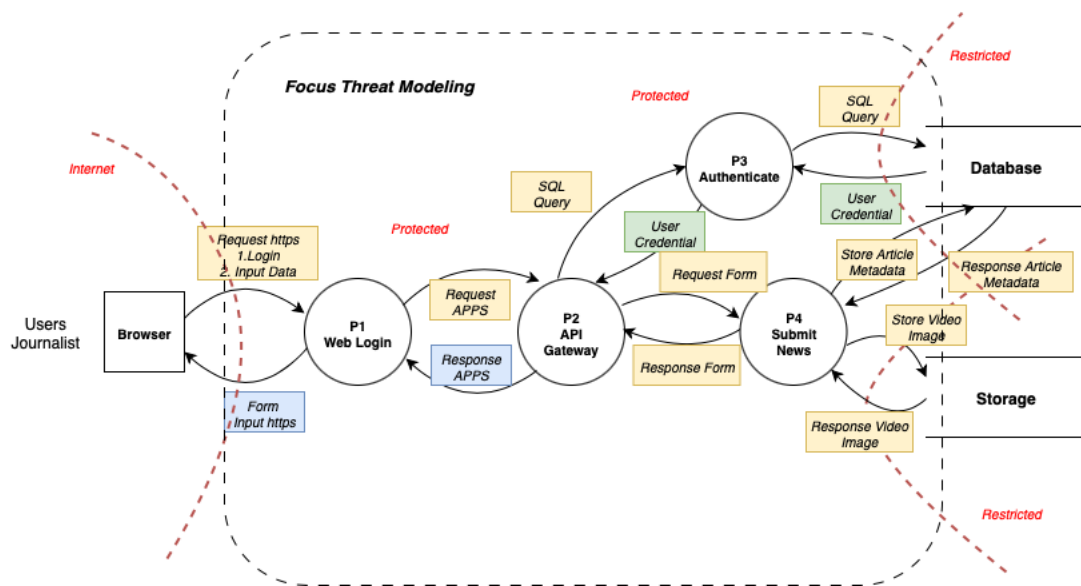


Figure 4.3. Data Flow Diagram Level 1 News Input Process.

Figure 4.3 above shows a data flow diagram for the process of **journalists inputting news content into the CMS** application, with 1 external entities, 4 processes, 14 data flows, and 2 data stores. This research has 4 asset components: User Credential Data, Article Data, Image Data, Video Data, External Data. There are 2 threat actors: Malicious users and Insiders.

In the context of a Content Management System (CMS), the process of collecting information assets based on taxonomy references from ISO/IEC 27005 involves systematically identifying and categorizing data elements that are critical to the organization's operations. ISO/IEC 27005 defines information assets as any data or information that has value to the organization and requires protection. Within a CMS, these assets typically include published articles, multimedia content, metadata, configuration files, user-generated data, and database records.

By mapping CMS components such as content repositories, user databases, and system logs to these taxonomy categories, organizations can ensure a comprehensive inventory of information assets. This structured approach enhances the ability to assess confidentiality, integrity, and availability (CIA) requirements for each asset, supporting

more effective threat modeling and risk management strategies. **Table 4.4** show is mapping asset taxonomy ISO 27005 of News Input Process CMS.

Table 4.4.Mapping Asset Taxonomy News Input Process.

Asset ID	Asset Name	Asset Type	ISO 27005 Asset Category	Process CMS	Description
A.01	Journalist Credential	Authentication Information	Information Asset	Storing Journalist Profile Information	Sensitive login credentials belonging to content journalists; requires strong confidentiality and integrity controls.
A.02	Database Content	Operational Data / Content Data	Information Asset	Make all news articles, and save images, videos.	Core content managed by CMS (articles, posts, media), considered high-value for the organization.
A.03	Database Metadata	Structural Metadata	Information Asset	Save information about content, date of manufacture, writer, category, tag.	Data describing other data, such as timestamps, user activity, content categories, tags. Crucial for search, analytics, and auditability.
A.04	Database Konfigurasi	System Configuration Asset	Information Asset	Storing system settings and user preferences.	Stores settings and environment variables for database operation such as access control, performance tuning, essential for availability and integrity.

Based on Figure 4.3 Input News Process DFD and asset identify that process, there are 4 processes and 3 component that occur as follows:

Table 4.5.Journalist News Input Process.

DFD Journalist		
Process ID	Process	Description
P.01	Web Login	Users enter Credencial data such as the user and login password and forwarded by the Balancer to API Gateway.
P.02	API Gateway	API Gateway processes the request to display forms to the user and save content to the database and storage.
P.03	Authenticate	User Credential process.
P.04	Submit News	As a place to input articles.
Component ID	Element	Description
EE.01	User Browser	The user browser represents the client-side interface used by end-users such as journalists, editors, and readers to interact with the CMS. It initiates requests for login, content creation, editing, publishing, and viewing published articles. DFD Role is external entity (EE1).
DS.01	Database	The database stores structured data that supports CMS operations. It contains user credentials, articles, post metadata, roles and permissions, session tokens, and configuration data. It is the core of the content management backend. DFD Role Data Store (DS1).
DS.02	Storage	Storage refers to the file system or object storage used for managing unstructured assets like images, videos, documents, or PDFs uploaded by users. It supports media delivery and links to content in the database.DFD Role Data Store (DS2).

Table 4.6 shows a detailed mapping of process items and processes for Journalist Input News to CMS, divided into 4 processes as follows:

Table 4.6. News Input Process Detail.

Items	Process Name	Detail Process
P.01	Web Login	1. Login form for user in the CMS 2. The user and password filling process to log into the application.
P.02	API Gateway	1. As an entry point for several services 2. Managing, authenticating, route fire call communication with external sources and send articles to platforms.
P.03	Authenticate	1. The process of sending and receiving user credentials from databases for users.
P.04	Submit News	1. As an input form for writing articles, uploading videos, uploading photos. 2. The process of adding articles from journalist coverage. 3. The process of seeing articles on journalist coverage results. 4. Content and metadata are stored in the database. 5. Video or photo is stored in storage.

Next, we'll describe the second process in the CMS: editors reviewing news content input by journalists or contributors before selecting it for submission to publishers. This process is illustrated in the Data Flow Diagram level 1 Figure 4.4. below.

4.3.3.2.Data Flow Stream News to Platform Process.

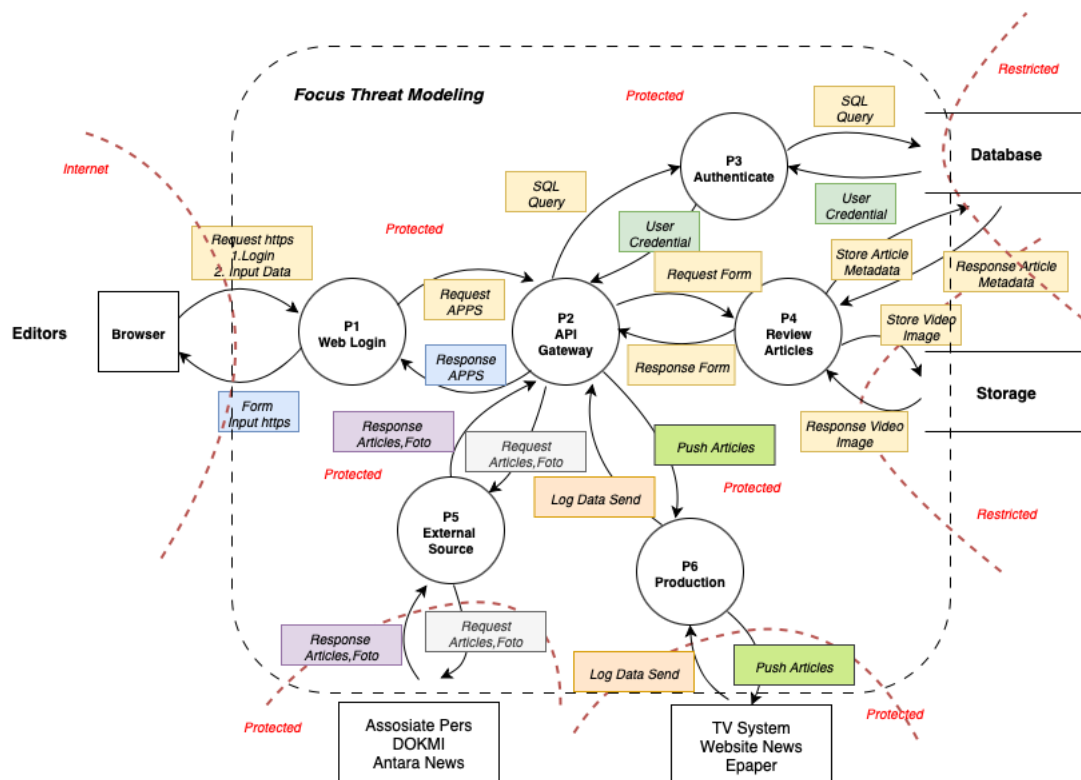


Figure 4.4. Data Flow Diagram Level 1 Stream News to Platform Process.

Figure 4.4 above shows a data flow diagram for the process of editor reviewing news content in the CMS application and publish to publisher, with **3** external entities, **6** processes, 22 data flows, and **2** data stores. This research has 4 asset components: User Credential Data, News Data, Image Data, Video Data, External Data, and there are 2 threat actors: Malicious users and Insiders.

After understanding the process in DFD figure 4.4, identify assets in the stream news to publish production process according to the ISO 27005 taxonomy as shown in table 4.7 below.

Table 4.7.List of Assets on Stream News to Platform Process.

Asset ID	Asset Name	Asset Type	ISO 27005 Asset Category	Process	Description
A.01	Editor Credential	Authentication Information	Information Asset	Storing Editor Profile Information	Sensitive login credentials belonging to content journalists; requires strong

Asset ID	Asset Name	Asset Type	ISO 27005 Asset Category	Process	Description
A.02	Database Content	Operational Data / Content Data	Information Asset	Make all news articles, and save images, videos.	confidentiality and integrity controls. Core content managed by CMS (articles, posts, media), considered high-value for the organization.
A.03	Database Metadata	Structural Metadata	Information Asset	Save information about content, date of manufacture, writer, category, tag.	Data describing other data, such as timestamps, user activity, content categories, tags. Crucial for search, analytics, and auditability.
A.04	Database Konfigurasi	System Configuration Asset	Information Asset	Storing system settings and user preferences.	Stores settings and environment variables for database operation such as access control, performance tuning, essential for availability and integrity.
A.05	Data Publish	Published Information	Information Asset	Data (Article, images,videos) publish to production such as television system, e-paper, and website.	Final rendered and published articles, pages, or media as delivered to end-users. Often replicates internal content with public access rights.

Table 4.8 explains the processes that occur in the editor and the components related to the process based on the data flow diagram in Figure 4.4 where there are 6 processes and 3 elements as in Table 4.8 as follows:

Table 4.8.Stream News to Production Process.

DFD Editor		
Process ID	Process	Description
P.01	Web Login	Users enter Credencial data such as the user and login password and forwarded by the Balancer to API Gateway.
P.02	API Gateway	API Gateway processes requests to display news lists to users, Edit Content features and edit results can be stored to databases and storage.
P.03	Authenticate	User Credential process.
P.04	Review Article	The results of the coverage can be edited in this section.
P.05	External Resource	Videos and images on External Source can be added to the edited content.
P.06	Stream Content to Platform.	Content that has been selected or edited can be forwarded to production platforms such as Epaper, TV, Portal News Website.
Component ID	Element	Description
EE.01	User Browser	The user browser represents the client-side interface used by end-users such as journalists, editors, and readers to interact with the CMS. It initiates requests for login, content creation, editing, publishing, and viewing published articles. DFD Role is external entity (EE1).
DS.01	Database	The database stores structured data that supports CMS operations. It contains user credentials, articles, post metadata, roles and permissions, session tokens, and configuration data. It is the core of the content management backend. DFD Role Data Store (DS1).
DS.02	Storage	Storage refers to the file system or object storage used for managing unstructured assets like images, videos, documents, or PDFs uploaded by users. It

DFD Editor		
Process ID	Process	Description
		supports media delivery and links to content in the database.DFD Role Data Store (DS2)

Table 4.9 shows the details of the editor's process, where these processes show what actions the editor takes until the process is complete, such as the login process, the process of viewing news resulting from journalist coverage, the process of editing news to the process of publishing news to the platform. The following detailed process in the editor process is explained in Table 4.9

Table 4.9. Stream News to Platform Process Detail

Items	Process Name	Detail Process
P.01	Web Login	1.Login form for user in the CMS 2. The user and password filling process to login to the application.
P.02	API Gateway	1. As an entry point for several services 2. Managing, authenticating, route fire call communication with external sources and send articles to platforms. 3. The process of storing articles and metadata to the database. 4. The process of storing videos or photos into storage.
P.03	Authenticate	1. The process of sending and receiving user credentials from databases for users.
P.04	Review Article	1. As a form to view and edit articles, upload videos, upload photos. 2. The process of seeing articles from journalist coverage. 3. The process of editing articles on journalist coverage results.

Items	Process Name	Detail Process
P.05	External Resource	1. As an external resource, the availability of sources such as photos, articles from third parties (AP, Antara, Dokmi) 2. The process of taking resources from third parties to edit articles on journalist coverage.
P.06	Production	1. The process of the results of the coverage that has been selected and edited is sent to platforms such as TV System, Website, Epaper.

According to Table 4.5 and Table 4.8, the entry point and trust level in the CMS system can be described, which can be useful in helping to determine critical points in the system and identify threats. The trust level determines who/what is trusted and to what extent, while the trust boundary shows the point of change of trust; the main place of security control.

4.3.4.Entry Point and Trust Level

In a general overview of a CMS system's architecture, there are typically designated points for both external and internal interactions. Externally, users access the system through a secured entry point, commonly a login interface, ensuring authenticated entry into the application.

Additionally, the system often interacts with third-party services and external networks, which are managed through a centralized gateway that regulates data exchange and maintains security protocols. Internally, there are entry points reserved for operational staff, accessible through a protected network channel, often involving additional security measures such as VPNs. These internal and external entry points are fundamental to the system's operation,

Trust levels in CMS classify system components, namely external entities, processes, and data stores based on the level of trust and vulnerability to security threats. This classification helps identify critical areas that require special protection. RBAC Users utilizes the risk identification of this trust level to define access control policies. On figure 4.4 explain data flow diagram trust level of CMS.

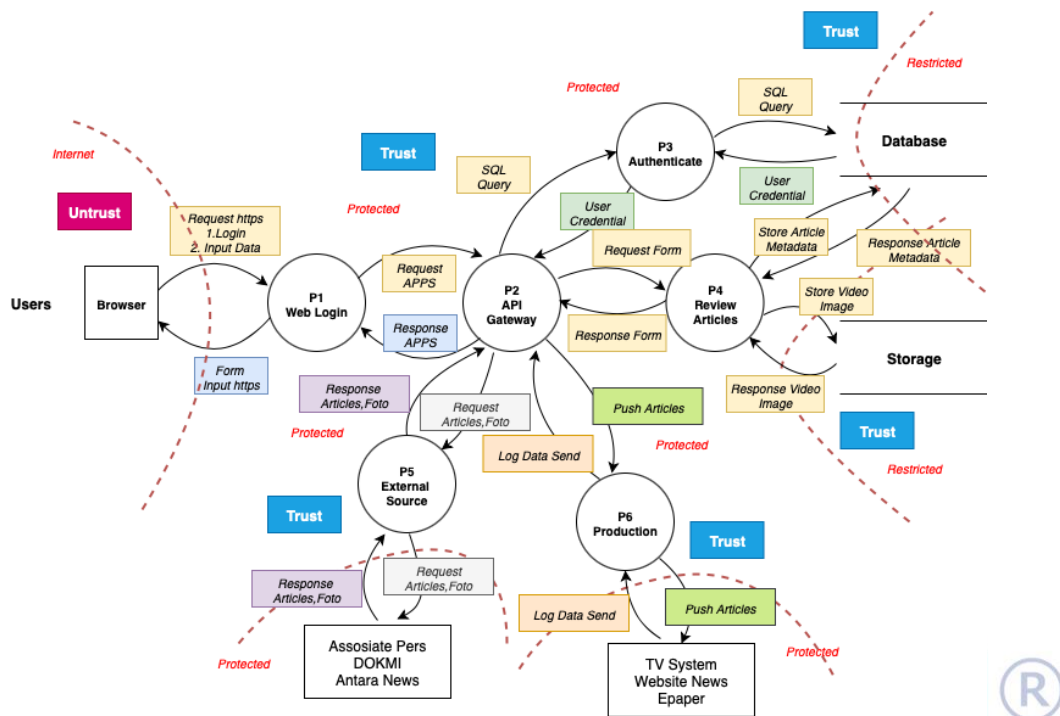


Figure 4.5. Data Flow Diagram Trust Level and Entry Point

According to figure 4.5 trust level and boundary layer below. The following is a table 4.10 of entry points in the CMS application.

Table 4.10. Entry Point of CMS

Component	Entry Point	Boundary
User Browser	Indirect	Untrust
Web Login	Exposed Port 443	Untrust
API Gateway	Internal	Trust
External Resources	External	Untrust
Database	Internal	Trust
Storage	Internal	Trust
Production Stream	Internal	Trust

From the entry point that we have defined according to the diagram in Figure 4.5, we can map the trust level in each section as below:

Table 4.11. Trust Level of CMS

Component	Entry Point	Trust Level
User Browser	Indirect	Untrusted input
Web Login	Exposed Port 443	Low (Directly interacts with users (untrusted sources))
API Gateway	Internal	Medium (Strict access controls and authentication)
External Source	External	Medium (Strict to other source)
Database	Internal	High (Internal access only)
Storage	Internal	High (Internal access only)
Production Stream	Internal	High (Internal access only)

There are 2 levels, namely untrust and trust, where each part has levels such as low, medium and high. On trust level

Roles in RBAC (such as admin, user, auditor) are designed in accordance with the confidence limit specified in the DFD above. This trust level becomes an analytical foundation for designing an effective RBAC structure, ensuring that user access rights are strictly in harmony with the previously mapped system security zone. RBAC of CMS explained on table 4.5.

Table 4.12. RBAC Users of CMS

User	Role	Description
Journalist	Read, Write	Is a user to write news from the results of journalists in the field.
Editor	Read, Write, Update	As an editor who can process and edit articles, videos and images, is tasked with publishing articles to external platforms.
Admin	Read, Write, Update	Administrator System that manages User Credential, Log and Content Monitoring

In this thesis, the threat modeling carried out focuses only on journalist and editor users, where these two users play a role in carrying out operational processes in accordance with PT XYZ's business processes. This CMS application is in accordance with the business process, intended only for internal journalist users, namely employees or

journalists of PT XYZ company where these journalists are in different locations so that this application is published via the internet.

4.4.Risk Identification

Risk identification in threat modeling and risk management research for journalists' Content Management System (CMS) applications plays a fundamental role in ensuring the security and sustainability of the system. The main objective is to systematically identify, analyze, and prioritize potential threats that can affect the integrity, confidentiality, and availability of the system. This process involves mapping threats such as SQL injection, Cross-Site Scripting (XSS), and unauthorized access, as well as evaluating vulnerabilities in the CMS security architecture and policies ((Shostack, 2014).

By using the Threat modeling and vulnerability assessment framework, researchers can categorize risks based on their severity, making it easier to allocate resources for mitigation (Howard and Lipner, 2006).

In addition, risk identification also ensures compliance with regulations such as GDPR and ISO/IEC 27001, which are very relevant given the sensitivity of data managed by journalists. The results of this process not only strengthen the system's resilience to attacks but also provide an empirical basis for stakeholders in making decisions related to security. Thus, risk identification becomes a critical step in designing a CMS that is safe, reliable, and in accordance with achieving the security business goals of PT XYZ. At this stage, threat modeling and vulnerability assessment are carried out. Threat modeling is carried out with various models according to the framework in Figure 3.1.

4.4.1.Threat Modeling

After creating a Data Flow Diagram in **figure 4.3** and **figure 4.4** At this stage, the following are the results of identifying threat based on STRIDE category. Threat mapping STRIDE by Input News Process (journalist) and Stream News to Platform (Editors). The results can be seen in **Table 4.13** and **Table 4.14**. Table 4.13 show threat on journalist process and Table 4.14 show threat on editors process.

4.4.1.1.Mapping Threat STRIDE Input News Process.

Table 4.13 is Following Threat Mapping STRIDE Input News Process (Journalist) :

Table 4.13. STRIDE Input News Process

Process / Component	Threat	S	T	R	I	D	E
P.01 Web Login	1.Credential Guessing						
	2.Session Theft	X					
	3.SQL Injection		X				
	4.Brute Force Attack Login.					X	
P.02 API Gateway	5.Token Reply Attack						
	6.Bypass Authentication	X					
	7.SQL Injection						
	8.Cross-site Scripting (XSS)		X				
	9.Query Parameter Injection.						
	10.MITM				X		
P.03 Authenticate	11.Brute Force Attack	X					
	12.Credential Reuse Attack.						
	13.SQL Injection		X				
	14. XSS Injection						
	15. MITM						
P.04 Submit News.	16.Session Hijacking	X					
	17. XSS Injection.		X				
	18.Malicious File Upload						
	19.Log manipulation			X			
	20.IDOR				X		
	21.DoS via large file uploads.					X	
EE.01 Web Browser	22.Session Hijacking	X					
DS.01 Database	23.Unauthorized DB Access				X		

Process / Component	Threat	S	T	R	I	D	E
DS.02 Storage	24. Exposed Storage Misconfiguration				X		

Note :

- 1.The symbol (X) explain that the threat with STRIDE category.
- 2.Detail MU form description in APPENDIX 2.

Table 4.13 explains the potential threat STRIDE categories in the process of journalists inputting news content into the CMS. Each process, **P.01 to P.04**, and each element **EE.01, DS.01,DS.02** , contains potential threats that could impact the CMS system.

4.4.1.2.Mapping Threat STRIDE Stream News to Platform.

The following threat mapping on the process stream to platform

Table 4.14. STRIDE Stream News to Production Process

Process /Component	Threat	S	T	R	I	D	E
P.01 Web Login	1.Credential Guessing						
	2.Session Theft	X					
	3.SQL Injection		X				
	4.Brute Force Attack Login.					X	
P.02 API Gateway	5.Token Reply Attack						
	6.Bypass Authentication	X					
	7.SQL Injection 8.Cross-site Scripting (XSS) 9.Query Parameter Injection.		X				
	10.MITM				X		
	11.Brute Force Attack	X					

Process /Component	Threat	S	T	R	I	D	E
P.03 Authenticate	12.Credential Reuse Attack.						
	13.SQL Injection						
	14. XSS Injection		X				
	15. MITM						
P.04 Review Articles	16.Session Hijacking	X					
	17. XSS Injection.						
	18.Malicious File Upload		X				
	19.Log manipulation			X			
	20.IDOR				X		
P.05 External Resource	21.DoS via large file uploads.					X	
	22.Spoofing	X					
	23.MITM		X				
	24.XSS				X		
P.06 Production	25.DoS					X	
	26. Misrouted Output Target	X					
	27.Content Injection		X				
	28. No Logging of Publish event			X			
	29.DoS					X	
EE.01 Web Browser	30.Privilege Escalation						X
	31.Session Hijacking	X					
DS.01 Database	32.Unauthorized DB Access				X		
DS.02 Storage	33.Exposed Storage Misconfiguration				X		

Note :

- 1.The symbol (X) explain that the potential threat with STRIDE Category.
- 2.Detail MU form description in APPENDIX 2

Table 4.13 explains the potential threat STRIDE categories in the process of editors review and send news content to publisher such as television system, website or e-paper. Each process, **P.01 to P.06**, and each element **EE.01, DS.01,DS.02** , contains potential threats that could impact the CMS system.

Furthermore, based on Table 4.4 and Table 4.7 regarding assets in the journalistic and editorial process which are very important, it is necessary to know the threats that occur, such as data theft or changes to assets with the privacy threat model.

4.4.2.Privacy Threat Modeling.

To find out the threats to privacy, it can be explained using the DFD news input process and stream news to publishing. Based on **Table 4.5** on RBAC and **Table 4.3**. can be described as the following authentication process. This is data flow diagram authentication and privacy od CMS.

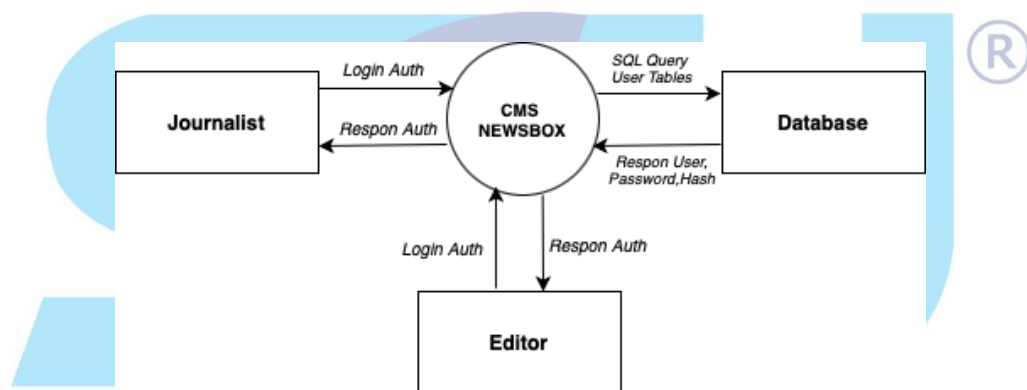
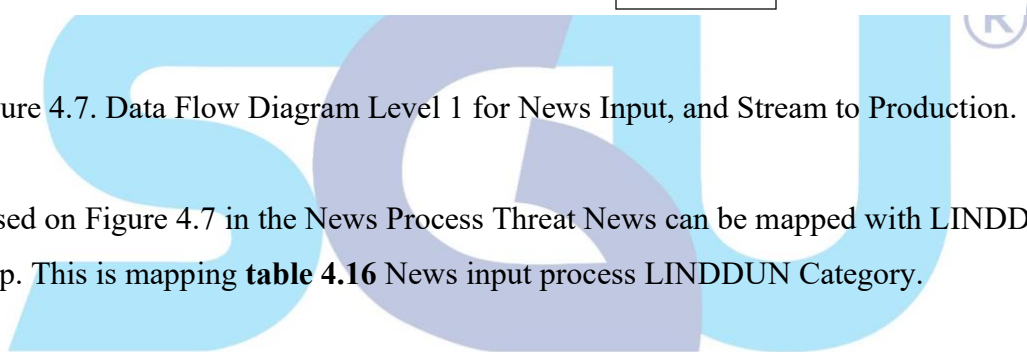


Figure 4.6. Data Flow Diagram Level 0 for Privacy, News Input and Edit Process.

From Figure 4.6, the following elements can be described in **Table 4.15**.

Table 4.15. DFD Element for Privacy

DFD Element	Description
Login Flow	Email and password can be recognized or leaked.
Data Stores : Users	DB user access without audit, does not match regulations.
Data Stores : Articles, logs	Content Articles in database.
Data Stores : Video, Images	Video and Images in storage
Transaction Flow	User transactions in the system.



Based on Figure 4.7 in the News Process Threat News can be mapped with LINDDUN Step. This is mapping **table 4.16** News input process LINDDUN Category.

The following threat mapping on the process input show in table 4.16;

1.Session Linkability	X						
-----------------------	---	--	--	--	--	--	--

Process/Component	Threat	L	I	N	D	D	U	N
P.01	1.Session Linkability	X						
P.02	2.Unauthorized API Access.						X	
P.03	3.Detectable Gateway Behavior-Scanning.				X			
P.04	4.Unauthorized Publishing						X	
EE.01	5.Browser Fingerprinting	X						
DS.01	6.Unauthorized Profile Access			X				
DS.02	7.Lack of Access Logging			X				

Note :

- 1.The symbol (X) explain that the potential threat with LINDDUN Category.
- 2.Detail MU Form description in APPENDIX 3.

Table 4.16 explains the potential threat LINDDUN categories in the process of journalists inputting news content into the CMS. Each process, **P.01 to P.04**, and each element **EE.01, DS.01,DS.02** , contains potential threats that could impact the CMS system.

4.4.2.2.Mapping Threat LINDDUN Stream News to Platform

Based on Figure 4.5 on stream to platform process threat can be mapped with LINDDUN category in **Table 4.17**.

Table. 4.17. Mapping Threat Model News Input to LINDDUN for Stream to Production Process.

Process/Component	Threat	L	I	N	D	D	U	N
P.01	1.Session Linkability	X						
P.02	2.Unauthorized API Access.						X	
P.03	3.Detectable Gateway Behavior-Scanning.				X			
P.04	4.Unauthorized Publishing						X	
P.05	5.MITM						X	
P.06	6.Publishing Attribution Denial.			X				
EE.01	7.Browser Fingerprinting	X						
DS.01	8.Unauthorized Profile Access			X				
DS.02	9.Lack of Access Logging			X				

Note :

- 1.The symbol (X) explain that the potential threat with LINDDUN Category.
- 2.Detail MU Form description in APPENDIX 3

Table 4.17 explains the potential threat LINDDUN categories in the process of editors review and send news content to publisher such as television system, website or e-paper.

Each process, **P.01 to P.06**, and each element **EE.01, DS.01,DS.02** , contains potential threats that could impact the CMS system.

4.3.Vulnerability Analysis.

At this stage, the following are the results of identifying weaknesses in the CMS application. Based on group discussion forums with developers, and IT Security, Misuse Case Forms can be seen in Appendix 1 and result scan vulnerability OWASP tools online obtained as follows :

Table 4.18. Mapping Threat and Vulnerability Input News Process.

No	Process/Component	Threat	Vulnerability	CVE
1	P.01 Web Login	Credential Guessing	No restrictions on API login attempts enables brute-force attacks to bypass authentication	CVE-2024-51558
2		Session Theft	Reflected XSS vulnerability allows attackers to steal authentication cookies and hijack sessions	CVE-2025-27088
3		SQL Injection	Manipulating SQL queries via unsanitized inputs	CVE-2025-52895 CVE-2025-8465
4		Brute Force Attack Login	Repeated OTP/password attempts accepted without checks.	CVE-2025-42600
5		Session Linkability	Session fixation weaknesses in web apps	CVE-2015-2295
6	P.02 API Gateway	Token Reply Attack	AuthContext tokens can be replayed to bypass login	CVE-2025-2230
7		Bypass Authentication	Authentication bypass vulnerability on web management interface	CVE-2025-0108
8		SQL Injection	Unsanitized inputs (e.g., query params) allow database compromise	CVE-2025-52895
9		Cross-site Scripting (XSS)	Stored XSS via input fields such as article title/body	CVE-2024-45856
10		Query Parameter Injection.	Malicious query parameters lead to SQLi or unexpected backend behavior	CVE-2025-46828
11		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952
12		Unauthorized API Access.	Attackers exploit missing authentication or insufficient authorization in API endpoints.	CVE-2023-27482

No	Process/Component	Threat	Vulnerability	CVE
13	P.03.Authenticate	Brute Force Attack	Attackers guess user credentials via repeated login attempts.	CVE-2019-17671.
14		Credential Reuse Attack.	Stolen credentials are used to log in to other accounts on the CMS, assuming password reuse.	CVE-2021-33564
15		SQL Injection	Unsanitized inputs (e.g., query params) allow database compromise	CVE-2025-52895
16		XSS Injection	Stored XSS via input fields such as article title/body	CVE-2024-45856
17		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952
18		Detectable Gateway Behavior-Scanning.	Attackers detect behaviors or responses of API gateway or reverse proxies to map the system and scan for weaknesses.	CVE-2022-26377
19	P.04. Submit News	Session Hijacking	Active user sessions are stolen via token capture.	CVE-2022-31793
20		XSS Injection	Stored XSS via input fields such as article title/body	CVE-2024-45856
21		Malicious File Upload	Attackers upload executable or script files disguised as media to gain remote access or execute code.	CVE-2021-24155
22		Log manipulation	Logs are deleted or altered by attackers to cover tracks or inject false data.	CVE-2022-34747
23		IDOR	Unauthorized access to internal objects (files, user profiles, media) by modifying URLs or IDs.	CVE-2022-2633
24		DoS via large file uploads.	Attackers upload oversized files, exhausting disk or memory resources, causing system unavailability.	CVE-2019-14422
25		Unauthorized Publishing	Attackers bypass access controls and publish unauthorized or malicious content.	CVE-2022-1096
26	EE.01 Browser	Session Hijacking	Active user sessions are stolen via token capture.	CVE-2022-31793
27		Browser Fingerprinting	Scripts in CMS track users by collecting device/browser features to uniquely identify them, often without consent.	CVE-2020-13942

No	Process/Component	Threat	Vulnerability	CVE
28	DS.01 Database	Unauthorized DB Access.	Attackers directly access or query the CMS database, extracting sensitive content or user data.	CVE-2022-0543
29		Unauthorized Profile Access	Attackers view or modify user profiles without permission via broken access control.	CVE-2022-21661
30	DS.02 Storage	Exposed Storage Misconfiguration	CMS media or backups are exposing content to the public.	CVE-2019-19781
31		Lack of Access Logging.	Admins cannot trace who accessed or modified CMS data due to missing or incomplete logs.	CVE-2021-4104

Table 4.19. Mapping Threat and Vulnerability Stream News to Production.

No	Process/Component	Threat	Vulnerability	CVE
1	P.01 Web Login	Credential Guessing	No restrictions on API login attempts enables brute-force attacks to bypass authentication	CVE-2024-51558
2		Session Theft	Reflected XSS vulnerability allows attackers to steal authentication cookies and hijack sessions	CVE-2025-27088
3		SQL Injection	Manipulating SQL queries via unsanitized inputs	CVE-2025-52895 CVE-2025-8465
4		Brute Force Attack Login	Repeated OTP/password attempts accepted without checks.	CVE-2025-42600
5		Session Linkability	Session fixation weaknesses in web apps	CVE-2015-2295
6	P.02 API Gateway	Token Reply Attack	HttpContext tokens can be replayed to bypass login	CVE-2025-2230
7		Bypass Authentication	Authentication bypass vulnerability on web management interface	CVE-2025-0108
8		SQL Injection	Unsanitized inputs (e.g., query params) allow database compromise	CVE-2025-52895
9		Cross-site Scripting (XSS)	Stored XSS via input fields such as article title/body	CVE-2024-45856
10		Query Parameter Injection.	Malicious query parameters lead to SQLi or unexpected backend behavior	CVE-2025-46828

No	Process/Component	Threat	Vulnerability	CVE
11		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952
12		Unauthorized API Access.	Attackers exploit missing authentication or insufficient authorization in API endpoints.	CVE-2023-27482
13	P.03.Authenticate	Brute Force Attack	Attackers guess user credentials via repeated login attempts.	CVE-2019-17671.
14		Credential Reuse Attack.	Stolen credentials are used to log in to other accounts on the CMS, assuming password reuse.	CVE-2021-33564
15		SQL Injection	Unsanitized inputs (e.g., query params) allow database compromise	CVE-2025-52895
16		XSS Injection	Stored XSS via input fields such as article title/body	CVE-2024-45856
17		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952
18		Detectable Gateway Behavior-Scanning.	Attackers detect behaviors or responses of API gateway or reverse proxies to map the system and scan for weaknesses.	CVE-2022-26377
19	P.04. Review Articles	Session Hijacking	Active user sessions are stolen via token capture.	CVE-2022-31793
20		XSS Injection	Stored XSS via input fields such as article title/body	CVE-2024-45856
21		Malicious File Upload	Attackers upload executable or script files disguised as media to gain remote access or execute code.	CVE-2021-24155
22		Log manipulation	Logs are deleted or altered by attackers to cover tracks or inject false data.	CVE-2022-34747
23		IDOR	Unauthorized access to internal objects (files, user profiles, media) by modifying URLs or IDs.	CVE-2022-2633
24		DoS via large file uploads.	Attackers upload oversized files, exhausting disk or memory resources, causing system unavailability.	CVE-2019-14422
25		Unauthorized Publishing	Attackers bypass access controls and publish unauthorized or malicious content.	CVE-2022-1096

No	Process/Component	Threat	Vulnerability	CVE
26	P.05 External Resource	Spoofing	Attacker pretends to be another user (e.g., journalist or admin) by bypassing or forging authentication.	CVE-2022-27664
27		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952
28		XSS	Stored XSS via input fields such as article title/body	CVE-2024-45856
29		DoS	Attackers send a high volume of requests or large payloads	CVE-2022-31160
30	P.06 Production	Misrouted Output Target	Content is published or displayed to the wrong audience or location due to misconfigured routing, permissions, or logic errors.	CVE-2022-24785
31		Content Injection	Malicious users inject unauthorized or malformed content (e.g., JavaScript, fake news) into CMS articles or fields due to improper input validation	CVE-2022-1184
33		No Logging of Publish event	The CMS lacks audit logging of when, who, and what content was published or modified.	CVE-2021-4104
34		DoS	Attackers send a high volume of requests or large payloads	CVE-2022-31160
35		Privilege Escalation	A lower-privileged user (e.g., journalist) gains higher-level access (e.g., admin/editor) due to flawed access control or misconfiguration.	CVE-2023-22527
36		Publishing Attribution Denial.	A user publishes content but attribution (author info or logs) is missing or suppressed intentionally or due to CMS flaws.	CVE-2022-31160
37	EE.01 Browser	Session Hijacking	Active user sessions are stolen via token capture.	CVE-2022-31793
38		Browser Fingerprinting	Scripts in CMS track users by collecting device/browser features to uniquely identify them, often without consent.	CVE-2020-13942
39	DS.01 Database	Unauthorized DB Access.	Attackers directly access or query the CMS database, extracting sensitive content or user data.	CVE-2022-0543

No	Process/Component	Threat	Vulnerability	CVE
40		Unauthorized Profile Access	Attackers view or modify user profiles without permission via broken access control.	CVE-2022-21661
41	DS.02 Storage	Exposed Storage Misconfiguration	CMS media or backups are exposing content to the public.	CVE-2019-19781
42		Lack of Access Logging.	Admins cannot trace who accessed or modified CMS data due to missing or incomplete logs.	CVE-2021-4104

Table 4.18 and table 4.19 explain the vulnerabilities found in the journalist and editor processes based on threats to all components mapped with CVE. After identifying the threats and vulnerabilities, the next step is to identify the security gaps to determine the current security condition of the application, along with the OWASP Top 10.

4.4. Security Gap Analysis

At this stage, the following are the results of security gaps in the CMS application. The results of the vulnerability analysis in **Table 4.18** can reveal the security gap or current condition of the application and environment. The results can be displayed in **Table 4.20** as follows:

Table 4.20. Security Gaps on Input News Process.

No	Process /Component	Threat	Vulnerability	CVE	OWASP	Security Gap
1	P.01 Web Login	Credential Guessing	No restrictions on API login attempts enables brute-force attacks to bypass authentication	CVE-2024-51558	A07: Identification & Authentication Failures	No rate limiting. No Captha
2		Session Theft	Reflected XSS vulnerability allows attackers to steal authentication cookies and hijack sessions	CVE-2025-27088	A07: Identification & Authentication Failures	Session ID exposed.
3		SQL Injection	Manipulating SQL queries via unsanitized inputs	CVE-2025-52895 CVE-2025-8465	A03: Injection	Possible to Direct injection of SQL via user input
4		Brute Force Attack Login	Repeated OTP/password attempts accepted without checks.	CVE-2025-42600	A07: Identification & Authentication Failures	No CAPTCHA / rate limit on login endpoint
5		Session Linkability	Session fixation weaknesses in web apps	CVE-2015-2295	A01: Broken Access Control	Static or predictable session identifiers

No	Process /Component	Threat	Vulnerability	CVE	OWASP	Security Gap
6	P.02 API Gateway	Token Reply Attack	AuthContext tokens can be replayed to bypass login	CVE-2025-2230	A07: Identification & Authentication Failures	Possible API tokens are reused.
7		Bypass Authentication	Authentication bypass vulnerability on web management interface	CVE-2025-0108	A01: Broken Access Control	Possible Flawed logic in auth flow
8		SQL Injection	Unsanitized inputs (e.g., query params) allow database compromise	CVE-2025-52895	A03: Injection	Possible to injection of SQL via user input.
9		Cross-site Scripting (XSS)	Stored XSS via input fields such as article title/body	CVE-2024-45856	A03: Injection	Possible No output encoding
10		Query Parameter Injection.	Malicious query parameters lead to SQLi or unexpected backend behavior	CVE-2025-46828	A03: Injection	Like SQL injection
11		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952	A02: Cryptographic Failures	No TLS or invalid certificate validation
12		Unauthorized API Access.	Attackers exploit missing authentication or insufficient authorization in API endpoints.	CVE-2023-27482	A01: Broken Access Control	API not protected with auth or scope checks
13	P.03. Authenticate	Brute Force Attack	Attackers guess user credentials via repeated login attempts.	CVE-2019-17671.	A07: Identification & Authentication Failures	No rate limiting. No Captha
14		Credential Reuse Attack.	Stolen credentials are used to log in to other accounts on the CMS, assuming password reuse.	CVE-2021-33564	A07: Identification & Authentication Failures	No rate limiting. No Captha
15		SQL Injection	Unsanitized inputs (e.g., query params) allow database compromise	CVE-2025-52895	A03: Injection	Possible to injection of SQL via user input.
16		XSS Injection	Stored XSS via input fields such as article title/body	CVE-2024-45856	A03: Injection	Possible No output encoding
17		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952	A02: Cryptographic Failures	No TLS or invalid certificate validation
18		Detectable Gateway Behavior-Scanning.	Attackers detect behaviors or responses of API gateway or reverse proxies to map the system and scan for weaknesses.	CVE-2022-26377	A06: Vulnerable & Outdated Components	API gateway reveals behavior via error messages
19		Session Hijacking	Active user sessions are stolen via token capture.	CVE-2022-31793	A07: Identification & Authentication Failures	Insecure session tokens or cookie theft

No	Process /Component	Threat	Vulnerability	CVE	OWASP	Security Gap
20	P.04. Submit News	XSS Injection	Stored XSS via input fields such as article title/body	CVE-2024-45856	A03: Injection	Possible No output encoding
21		Malicious File Upload	Attackers upload executable or script files disguised as media to gain remote access or execute code.	CVE-2021-24155	A05: Security Misconfiguration	No file type or MIME check; no sandboxing
22		Log manipulation	Logs are deleted or altered by attackers to cover tracks or inject false data.	CVE-2022-34747	A09: Security Logging and Monitoring Failures	Logs not integrity-protected or stored securely
23		IDOR	Unauthorized access to internal objects (files, user profiles, media) by modifying URLs or IDs.	CVE-2022-2633	A01: Broken Access Control	Possible No access control check on object/resource ID
24		DoS via large file uploads.	Attackers upload oversized files, exhausting disk or memory resources, causing system unavailability.	CVE-2019-14422	A01: Broken Access Control	No size restriction or quota limit on media
25		Unauthorized Publishing	Attackers bypass access controls and publish unauthorized or malicious content.	CVE-2022-1096	A01: Broken Access Control	No role checks or flawed publish permission logic
26	EE.01 Browser	Session Hijacking	Active user sessions are stolen via token capture.	CVE-2022-31793	A07: Identification & Authentication Failures	Insecure session tokens or cookie theft
27		Browser Fingerprinting	Scripts in CMS track users by collecting device/browser features to uniquely identify them, often without consent.	CVE-2020-13942	A01: Broken Access Control	No randomization, no privacy headers
28	DS.01 Database	Unauthorized DB Access.	Attackers directly access or query the CMS database, extracting sensitive content or user data.	CVE-2022-0543	A01: Broken Access Control	Misconfiguration, port can exposed to internal network.
29		Unauthorized Profile Access	Attackers view or modify user profiles without permission via broken access control.	CVE-2022-21661	A01: Broken Access Control	No access check for profile read/update
30	DS.02 Storage	Exposed Storage Misconfiguration	CMS media or backups are exposing content to the public.	CVE-2019-19781	A05: Security Misconfiguration	Possible to Poor config leads to data leakage
31		Lack of Access Logging.	Admins cannot trace who accessed or modified CMS data due to missing or incomplete logs.	CVE-2021-4104	A09: Security Logging and Monitoring Failures	CMS lacks audit trail for content and user access

The results of the vulnerability analysis in **Table 4.19** can reveal the security gap or current condition of the application and environment. The results can be displayed in **Table 4.21** as follows:

Table 4.21. Security Gap on Stream News to Platform Process.

No	Process /Component	Threat	Vulnerability	CVE	OWASP	Security Gap
1	P.01 Web Login	Credential Guessing	No restrictions on API login attempts enables brute-force attacks to bypass authentication	CVE-2024-51558	A07: Identification & Authentication Failures	No rate limiting. No Captha
2		Session Theft	Reflected XSS vulnerability allows attackers to steal authentication cookies and hijack sessions	CVE-2025-27088	A07: Identification & Authentication Failures	Session ID exposed.
3		SQL Injection	Manipulating SQL queries via unsanitized inputs	CVE-2025-52895 CVE-2025-8465	A03: Injection	Possible to Direct injection of SQL via user input
4		Brute Force Attack Login	Repeated OTP/password attempts accepted without checks.	CVE-2025-42600	A07: Identification & Authentication Failures	No CAPTCHA / rate limit on login endpoint
5		Session Linkability	Session fixation weaknesses in web apps	CVE-2015-2295	A01: Broken Access Control	Static or predictable session identifiers
6	P.02 API Gateway	Token Reply Attack	AuthContext tokens can be replayed to bypass login	CVE-2025-2230	A07: Identification & Authentication Failures	Possible API tokens are reused.
7		Bypass Authentication	Authentication bypass vulnerability on web management interface	CVE-2025-0108	A01: Broken Access Control	Possible Flawed logic in auth flow
8		SQL Injection	Unsanitized inputs (e.g., query params) allow database compromise	CVE-2025-52895	A03: Injection	Possible to injection of SQL via user input.
9		Cross-site Scripting (XSS)	Stored XSS via input fields such as article title/body	CVE-2024-45856	A03: Injection	Possible No output encoding
10		Query Parameter Injection.	Malicious query parameters lead to SQLi or unexpected backend behavior	CVE-2025-46828	A03: Injection	Like SQL injection
11		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952	A02: Cryptographic Failures	No TLS or invalid certificate validation
12		Unauthorized API Access.	Attackers exploit missing authentication or insufficient authorization in API endpoints.	CVE-2023-27482	A01: Broken Access Control	API not protected with auth or scope checks

No	Process /Component	Threat	Vulnerability	CVE	OWASP	Security Gap
13	P.03 .Authenticate	Brute Force Attack	Attackers guess user credentials via repeated login attempts.	CVE-2019-17671.	A07: Identification & Authentication Failures	No rate limiting. No Captha
14		Credential Reuse Attack.	Stolen credentials are used to log in to other accounts on the CMS, assuming password reuse.	CVE-2021-33564	A07: Identification & Authentication Failures	No rate limiting. No Captha
15		SQL Injection	Unsanitized inputs (e.g., query params) allow database compromise	CVE-2025-52895	A03: Injection	Possible to injection of SQL via user input.
16		XSS Injection	Stored XSS via input fields such as article title/body	CVE-2024-45856	A03: Injection	Possible No output encoding
17		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952	A02: Cryptographic Failures	No TLS or invalid certificate validation
18		Detectable Gateway Behavior-Scanning.	Attackers detect behaviors or responses of API gateway or reverse proxies to map the system and scan for weaknesses.	CVE-2022-26377	A06: Vulnerable & Outdated Components	API gateway reveals behavior via error messages
19	P.04. Review Articles	Session Hijacking	Active user sessions are stolen via token capture.	CVE-2022-31793	A07: Identification & Authentication Failures	Insecure session tokens or cookie theft
20		XSS Injection	Stored XSS via input fields such as article title/body	CVE-2024-45856	A03: Injection	Possible No output encoding
21		Malicious File Upload	Attackers upload executable or script files disguised as media to gain remote access or execute code.	CVE-2021-24155	A05: Security Misconfiguration	No file type or MIME check; no sandboxing
22		Log manipulation	Logs are deleted or altered by attackers to cover tracks or inject false data.	CVE-2022-34747	A09: Security Logging and Monitoring Failures	Logs not integrity-protected or stored securely
23		IDOR	Unauthorized access to internal objects (files, user profiles, media) by modifying URLs or IDs.	CVE-2022-2633	A01: Broken Access Control	Possible No access control check on object/resource ID
24		DoS via large file uploads.	Attackers upload oversized files, exhausting disk or memory resources, causing system unavailability.	CVE-2019-14422	A01: Broken Access Control	No size restriction or quota limit on media
25		Unauthorized Publishing	Attackers bypass access controls and publish unauthorized or malicious content.	CVE-2022-1096	A01: Broken Access Control	No role checks or flawed publish permission logic
26		Spoofing	Attacker pretends to be another user (e.g., journalist	CVE-2022-27664		

No	Process /Component	Threat	Vulnerability	CVE	OWASP	Security Gap
			or admin) by bypassing or forging authentication.			
27		MITM	An attacker intercepts communications between the user and the server to steal credentials or modify data.	CVE-2023-22952	A02: Cryptographic Failures	No TLS or invalid certificate validation
28		XSS	Stored XSS via input fields such as article title/body	CVE-2024-45856	A03: Injection	Possible No output encoding
29		DoS	Attackers send a high volume of requests or large payloads	CVE-2022-31160	A01: Broken Access Control	No size restriction or quota limit on media
30		Misrouted Output Target	Content is published or displayed to the wrong audience or location due to misconfigured routing, permissions, or logic errors.	CVE-2022-24785	A01: Broken Access Control	Insecure API response or misconfigured output channels
31		Content Injection	Malicious users inject unauthorized or malformed content (e.g., JavaScript, fake news) into CMS articles or fields due to improper input validation	CVE-2022-1184	A03: Injection	Possible to injection of SQL or XSS.
32		No Logging of Publish event	The CMS lacks audit logging of when, who, and what content was published or modified.	CVE-2021-4104	A09: Security Logging and Monitoring Failures	CMS lacks audit trail for content and user access
33		DoS	Attackers send a high volume of requests or large payloads	CVE-2022-31160	A01: Broken Access Control	No size restriction or quota limit on media
34		Privilege Escalation	A lower-privileged user (e.g., journalist) gains higher-level access (e.g., admin/editor) due to flawed access control or misconfiguration.	CVE-2023-22527	A01: Broken Access Control	Weak role validation
35		Publishing Attribution Denial.	A user publishes content but attribution (author info or logs) is missing or suppressed intentionally or due to CMS flaws.	CVE-2022-31160	A01: Broken Access Control	No size restriction or quota limit on media
36	EE.01 Browser	Session Hijacking	Active user sessions are stolen via token capture.	CVE-2022-31793	A07: Identification & Authentication Failures	Insecure session tokens or cookie theft
37		Browser Fingerprinting	Scripts in CMS track users by collecting device/browser features to uniquely identify them, often without consent.	CVE-2020-13942	A01: Broken Access Control	No randomization, no privacy headers
38		Unauthorized DB Access.	Attackers directly access or query the CMS database,	CVE-2022-0543	A01: Broken Access Control	Misconfiguration, port can exposed to internal network.

No	Process /Component	Threat	Vulnerability	CVE	OWASP	Security Gap
	DS.01 Database		extracting sensitive content or user data.			
39		Unauthorized Profile Access	Attackers view or modify user profiles without permission via broken access control.	CVE-2022-21661	A01: Broken Access Control	No access check for profile read/update
40	DS.02 Storage	Exposed Storage Misconfiguration	CMS media or backups are exposing content to the public.	CVE-2019-19781	A05: Security Misconfiguration	Possible to Poor config leads to data leakage
41		Lack of Access Logging.	Admins cannot trace who accessed or modified CMS data due to missing or incomplete logs.	CVE-2021-4104	A09: Security Logging and Monitoring Failures	CMS lacks audit trail for content and user access

After identifying threats, vulnerabilities, and security gaps within a Content Management System (CMS), this step aims to thoroughly understand potential security risks that could compromise the integrity, confidentiality, and availability of information managed by the CMS. Threats describe actors or events that could potentially attack the system, such as hackers, malware, or insider threats. Meanwhile, vulnerabilities indicate weak points in the system, such as those in **Table 4.18** and **Table 4.19**, such as unvalidated form input or insecure server configurations, which could be exploited by threats. Security gaps in **Table 4.20** and **Table 4.21** refer to the absence of security controls that should be implemented, such as the lack of two-factor authentication, logging, or an intrusion detection system.

Next, risks are identified using the DREAD model and prioritized using a risk scoring method. By systematically identifying all these elements, organizations can prioritize protection, develop appropriate mitigation strategies, and allocate security resources efficiently. As well as providing an understanding that supports the secure by design approach, namely building a secure CMS system from the design stage.

4.5.Risk Analysis

After indentifying threat and vulnerabilities in each component journalist and editors process, as present in **Table 4.20** and **Table 4.21**, the next step is to evaluate this finding using DREAD Method. This stage allows for determining the risk and prioritization of

these identified threats. The DREAD value of each threat in Table 4.22 is determining using Table 3.2 as a reference, and the results shown in Table 4.22 and Table 4.23.

4.5.1. Threat Scoring

The following are the threat scoring results for the Input News Process and Stream News to Production, which can be seen in detail in appendix 4 and appendix 7.

Table 4.22. Table Risk Scoring Input News Process.

No	Process	Threat	D	R	E	A	D	Score
1.	P.01	Credential Guessing	6	8	9	7	8	7,6
2.	Web Login	Session Theft	8	7	8	8	7	7,6
3.		SQL Injection	9	8	9	9	8	8,6
4.		Brute Force Attack Login	7	9	9	7	8	8
5.		Session Linkability	5	6	4	6	7	5,6
6.	P.02	Token Reply Attack	7	9	9	7	8	8
7.	API Gateway	Bypass Authentication	9	8	8	9	8	8,4
8.		SQL Injection	9	8	9	9	8	8,6
9.		Cross-site Scripting (XSS)	7	8	8	6	8	7,4
10.		Query Parameter Injection.	7	7	8	6	8	7,2
11.		MITM	8	9	7	6	6	7,2
12.		XSS	7	8	8	6	8	7,4
13.		Unauthorized API Access.	9	8	9	8	7	8,2
14.	P.03.Authenticate	Brute Force Attack	7	9	9	6	7	7,6
15.		Credential Reuse Attack.	8	9	8	7	7	7,8
16.		SQL Injection	9	8	9	9	8	8,6
17.		XSS Injection	6	7	7	5	7	6,4
18.		Detectable Gateway Behavior-Scanning.	5	6	6	5	7	5,8
19.	P.04.	Session Hijacking	8	7	8	7	7	7,4
20.	Submit News	XSS Injection	7	8	8	6	8	7,4
21.		Malicious File Upload	9	7	8	7	6	7,4
22.		Log manipulation	6	6	5	5	5	5,4
23.		IDOR	8	7	7	8	6	7,2
24.		DoS via large file uploads.	7	8	8	6	7	7,2
25.	EE.01 Browser	Unauthorized Publishing	9	7	6	9	6	7,4
26.		Session Hijacking	7	8	8	7	9	7,8
27.		Browser Fingerprinting	6	7	8	8	9	7,6

No	Process	Threat	D	R	E	A	D	Score
28.	DS.01 Database	Unauthorized DB Access.	8	7	8	9	7	7,8
29.		Unauthorized Profile Access	7	8	7	8	8	7,6
30.	DS.02 Storage	Exposed Storage	7	8	6	8	9	7,6
		Misconfiguration						
31.		Lack of Access Logging	5	6	6	6	5	5,6

Table 4.23. Mapping Threat Stream News to Production.

No	Process	Threat	D	R	E	A	D	Score
1.	P.01 Web Login	Credential Guessing	6	8	9	7	8	7,6
2.		Session Theft	8	7	8	8	7	7,6
3.		SQL Injection	9	8	9	9	8	8,6
4.		Brute Force Attack Login	7	9	9	7	8	8
5.		Session Linkability	5	6	4	6	7	5,6
6.	P.02	Token Reply Attack	7	9	9	7	8	8
7.	API Gateway	Bypass Authentication	9	8	8	9	8	8,4
8.		SQL Injection	9	8	9	9	8	8,6
9.		Cross-site Scripting (XSS)	7	8	8	6	8	7,4
10.		Query Parameter Injection.	7	7	8	6	8	7,2
11.		MITM	8	9	7	6	6	7,2
12.		XSS	7	8	8	6	8	7,4
13.		Unauthorized API Access.	9	8	9	8	7	8,2
14.	P.03.Authenticate	Brute Force Attack	7	9	9	6	7	7,6
15.		Credential Reuse Attack.	8	9	8	7	7	7,8
16.		SQL Injection	9	8	9	9	8	8,6
17.		XSS Injection	6	7	7	5	7	6,4
18.		Detectable Gateway Behavior-Scanning.	5	6	6	5	7	5,8
19.	P.04.	Session Hijacking	8	7	8	7	7	7,4
20.	Review Articles	XSS Injection	7	8	8	6	8	7,4
21.		Malicious File Upload	9	7	8	7	6	7,4
22.		Log manipulation	6	6	5	5	5	5,4
23.		IDOR	8	7	7	8	6	7,2

No	Process	Threat	D	R	E	A	D	Score
24.		DoS via large file uploads.	7	9	9	7	8	8
25.		Unauthorized Publishing	9	7	6	9	6	7,4
26.	P.05 External	Spoofing	8	8	7	7	6	7,2
27.	Resource	MITM	8	7	8	7	7	7,4
28.		XSS	6	8	8	6	8	7,2
29.		DoS	7	8	7	6	7	7
30.	P.06 Production	Misrouted Output Target	7	6	6	7	5	6,2
31.		Content Injection	8	8	7	8	7	7,6
32.		No Logging of Publish event	6	7	5	6	5	5,8
33.		DoS	7	8	7	7	7	7,2
34.		Privilege Escalation	9	7	7	9	6	7,6
35.		Publishing Attribution Denial.	6	6	6	5	6	5,8
36.	EE.01 Browser	Session Hijacking	7	8	8	7	9	7,8
37.		Browser Fingerprinting	6	7	8	8	9	7,6
38.	DS.01 Database	Unauthorized DB Access.	8	7	8	9	7	7,8
39.		Unauthorized Profile Access	7	8	7	8	8	7,6
40.	DS.02 Storage	Exposed Storage Misconfiguration	7	8	6	8	9	7,6
41.		Lack of Access Logging	5	6	6	6	5	5,6

4.5.2. Threat Prioritization

After identifying threats based on the scores in each component of the journalist and editor process, as presented in **Table 4.22** and **Table 4.23**, the next step is to prioritize threats based on the highest to lowest scores as seen in **Table 24** and **Table 25**.

Table 4.24. Threat Prioritization Input News Process

No	Process/Component	Threat	Risk Score	Risk Level
1.	P.01 Web Login	SQL Injection	8,6	High
2.	P.02 API Gateway	SQL Injection	8,6	High
3.	P.03 Authenticate	SQL Injection	8,6	High
4.	P.02 API Gateway	Bypass Authentication	8,4	High
5.	P.02 API Gateway	Unauthorized API Access.	8,2	High
6.	P.01 Web Login	Brute Force Attack Login	8	High

No	Process/Component	Threat	Risk Score	Risk Level
7.	P.02 API Gateway	Token Reply Attack	8	High
8.	P.03 Authenticate	Credential Reuse Attack.	7,8	High
9.	EE.01 Browser	Session Hijacking	7,8	High
10.	DS.01 Database	Unauthorized DB Access.	7,8	High
11.	P.01 Web Login	Credential Guessing	7,6	High
12.	P.01 Web Login	Session Theft	7,6	High
13.	P.03 Authenticate	Brute Force Attack	7,6	High
14.	EE.01 Browser	Browser Fingerprinting	7,6	High
15.	DS.01 Database	Unauthorized Profile Access	7,6	High
16.	DS.02 Storage	Exposed Storage Misconfiguration	7,6	High
17.	P.02 API Gateway	Cross-site Scripting (XSS)	7,4	High
18.	P.02 API Gateway	XSS	7,4	High
19.	P.04 Submit News	Session Hijacking	7,4	High
20.	P.04 Submit News	XSS Injection	7,4	High
21.	P.04 Submit News	Malicious File Upload	7,4	High
22.	P.04 Submit News	Unauthorized Publishing	7,4	High
23.	P.02 API Gateway	Query Parameter Injection.	7,2	High
24.	P.02 API Gateway	MITM	7,2	High
25.	P.04 Submit News	IDOR	7,2	High
26.	P.04 Submit News	DoS via large file uploads.	7,2	High
27.	P.03 Authenticate	XSS Injection	6,4	Medium
28.	P.03 Authenticate	Detectable Gateway Behavior-Scanning.	5,8	Medium
29.	P.01 Web Login	Session Linkability	5,6	Medium
30.	DS.02 Storage	Lack of Access Logging	5,6	Medium
31.	P.04 Submit News	Log manipulation	5,4	Medium

Table 4.25. Threat Prioritization Stream News to Production.

No	Process/Component	Threat	Risk Score	Risk Level
1.	P.01 Web Login	SQL Injection	8,6	High
2.	P.02 API Gateway	SQL Injection	8,6	High
3.	P.03.Authenticate	SQL Injection	8,6	High
4.	P.02 API Gateway	Bypass Authentication	8,4	High
5.	P.02 API Gateway	Unauthorized API Access.	8,2	High
6.	P.01 Web Login	Brute Force Attack Login	8	High
7.	P.02 API Gateway	Token Reply Attack	8	High
8.	P.04. Review Articles	DoS via large file uploads.	8	High
9.	P.03.Authenticate	Credential Reuse Attack.	7,8	High
10.	EE.01 Browser	Session Hijacking	7,8	High
11.	DS.01 Database	Unauthorized DB Access.	7,8	High
12.	P.01 Web Login	Credential Guessing	7,6	High

No	Process/Component	Threat	Risk Score	Risk Level
13.	P.01 Web Login	Session Theft	7,6	High
14.	P.03.Authenticate	Brute Force Attack	7,6	High
15.	P.06 Production	Content Injection	7,6	High
16.	P.06 Production	Privilege Escalation	7,6	High
17.	EE.01 Browser	Browser Fingerprinting	7,6	High
18.	DS.01 Database	Unauthorized Profile Access	7,6	High
19.	DS.02 Storage	Exposed Storage Misconfiguration	7,6	High
20.	P.02 API Gateway	Cross-site Scripting (XSS)	7,4	High
21.	P.02 API Gateway	XSS	7,4	High
22.	P.04. Review Articles	Session Hijacking	7,4	High
23.	P.04. Review Articles	XSS Injection	7,4	High
24.	P.04. Review Articles	Malicious File Upload	7,4	High
25.	P.04. Review Articles	Unauthorized Publishing	7,4	High
26.	P.05 External Resource	MITM	7,4	High
27.	P.02 API Gateway	Query Parameter Injection.	7,2	High
28.	P.02 API Gateway	MITM	7,2	High
29.	P.04. Review Articles	IDOR	7,2	High
30.	P.05 External Resource	Spoofing	7,2	High
31.	P.05 External Resource	XSS	7,2	High
32.	P.06 Production	DoS	7,2	High
33.	P.05 External Resource	DoS	7	High
34.	P.03.Authenticate	XSS Injection	6,4	Medium
35.	P.06 Production	Misrouted Output Target	6,2	Medium
36.	P.03.Authenticate	Detectable Gateway Behavior-Scanning.	5,8	Medium
37.	P.06 Production	No Logging of Publish event	5,8	Medium
38.	P.06 Production	Publishing Attribution Denial.	5,8	Medium
39.	P.01 Web Login	Session Linkability	5,6	Medium
40.	DS.02 Storage	Lack of Access Logging	5,6	Medium
41.	P.04. Review Articles	Log manipulation	5,4	Medium

Note :

High Risk (10-7)
Medium Risk (6-4)
Low Risk (3-0)



4.6.Determine Countermeasures

After obtaining the threat priority in each component of the journalist and editor process, as presented in **Table 4.24** and **Table 4.25**, the next step is to create a mitigation control by selecting **8 process** mitigation controls with an average **score of 8**. These mitigation

controls are based on OWASP Top 10 and ISO 27001:2022, the results can be seen in **Table 4.26** and **Table 4.27**.

4.6.1 Technical Mitigation.

The following mitigation controls for technical circles with the highest threat priority based on the 2021 OWASP Top 10 controls are shown in table 4.26 below:

Table 4.26. Recommendation Mitigation By OWASP Top 10

Threat	OWASP Top 10 (2021)	Mitigation	Recommendation	Risk
SQL Injection	A03:2021 – Injection	1.Use parameterized queries. 2.Validate and sanitize all user inputs. 3. In Process P.01,P.02,P.03	1.Avoid dynamic SQL. 2.Use ORM libraries. 3.Regularly test with automated scanners. 4. Evaluate secure code on P.01,P.02,P.03	High
Bypass Authentication	A01:2021 – Broken Access Control	1.Enforce strict access control checks server-side. 2.Do not expose authentication logic in the frontend. 3. In Process P.02	1.Implement Multi-Factor Authentication (MFA) 2.Use secure session tokens. 3. Evaluate secure code on P.02	High
Unauthorized API Access	A01:2021 – Broken Access Control	1.Implement role-based access control (RBAC). 2.Enforce authentication and authorization at each endpoint. 3. In Process P.02	1.Use API gateways with strict access control policies. 2.Token validation on each request. 3. Evaluate secure code on P.02.	High
Token Replay Attack	A07:2021 – Identification & Auth Failures	1.Use short-lived access tokens 2.Implement token binding. 3.Enable token expiration and revocation lists. 4. In Process P.02	1.Use nonce or timestamp in token 2.Consider OAuth2.0 with PKCE. 3.Implement TLS everywhere. 4. Evaluate secure code on P.02.	High
DoS via large file uploads.	A05:2021 – Security Misconfiguration / A01:2021 – Broken Access Control	1.Limit file upload size 2.Use content-type checking 3.Implement upload rate limits 4.Store files outside of app directory. 5.In Process P.04	1.Enforce quotas per user 2.Use virus/malware scanners 3.Set HTTP timeout and max payload limits on reverse proxies. 4. Evaluate secure code on P.04.	High
Credential Reuse Attack	A07:2021 – Identification & Auth Failures	1.Detect known leaked credentials. 2.Enforce password uniqueness and rotation. 3. In Process P.03	1.Implement MFA 2.Block known breached passwords 3.Educate users about password hygiene. 4. Evaluate secure code on P.03.	High

4.6.2 Non Technical Mitigation.

Here is a refined threat-mitigation table that focuses specifically on the organizational and management measures from ISO/IEC 27001:2022, particularly Clause 5: Organizational controls, for managing web CMS showing in Table 4.27 below :

Table 4.27. Recommendation Mitigation By ISO 27001:2022

Threat	ISO 27001:2022	Management Mitigation	Recommendation	Risk
SQL Injection	5.36 Application security testing 5.25 Secure development lifecycle	1.Enforce secure coding policies and developer training 2.Include injection risks in secure SDLC policy. 3.Perform regular secure code reviews	1.Establish a secure development lifecycle (SDLC) policy 2.Mandatory OWASP training for developers	High
Bypass Authentication	5.15 Access rights 5.17 Password management	1.Enforce access control policy 2.Define roles and responsibilities clearly 3.Ensure change management on auth modules	1.Implement centralized IAM (Identity and Access Management) 2.Enforce peer review on changes	High
Unauthorized API Access	5.14 API security 5.16 Access restrictions	1.Define API security policy 2.Set third-party integration approval and API usage governance 3.Log and audit API access by policy	Define API classification levels (public/internal/partner) and security expectations.	High
Token Replay Attack	5.18 Session timeout and termination 5.36 Security testing	1.Set policies on token lifecycle management 2.Require secure transmission policies (TLS-only channels) 3.Conduct periodic auth reviews	1.Include token attack scenarios in threat models 2.Periodically review authentication mechanisms	High
DoS via large file uploads.	5.17 Password policy 5.11 User awareness	1.Policy to prohibit password reuse. 2.Enforce use of credential managers. 3.Educate users on reuse risks.	1.Integrate credential checking APIs with breach databases. 2.Set MFA as mandatory	High
Credential Reuse Attack	5.23 Use of cloud services 5.31 Capacity management	1.Policy for max file size & type limits 2.Use upload governance for source control. 3.Define response plan for service overload	1.Add WAF rate limiting policy 2.Define acceptable file formats and size thresholds in information security policy	High

Security threats, such as those described in Tables 4.26 and 4.27, such as SQL Injection, Bypass Authentication, Unauthorized API Access, Token Replay Attacks, and Denial of Service (DoS) attacks through large file uploads, pose serious risks to the confidentiality, integrity, and availability of personal data. To mitigate these risks, company management needs to adopt a comprehensive approach encompassing technical aspects, policies, and compliance with the Personal Data Protection Law (Law No. 27 of 2022). From a technical perspective, implementing secure coding practices, multi-factor authentication, API security protocols, short-lived tokens, and file upload restrictions are crucial steps to mitigate potential attacks. From a policy perspective, implementing identity and access management standards, vendor risk management, and internal security policies that bind all stakeholders is necessary. Meanwhile, from a regulatory perspective, the 2022 Personal Data Protection Law requires data controllers to protect personal data (Pasal 35), limit unauthorized access (Pasal 39), and ensure accountability and transparency in data management (Pasal 51). Therefore, management is not only responsible for providing technological solutions, but is also obliged to ensure risk governance, compliance audits, and incident response plan readiness so that these threats do not result in legal losses or damage to the company's reputation.

Company management must view the above mitigation measures not only from a technical perspective, but also as a legal obligation to comply with the 2022 Privacy and Security Law (UU PDP, 2022). The following are cybersecurity awareness programs that management can implement:

1. Investment in technology security (AST, IAM, API security, anti-DDoS).
2. Implementation of internal policies consistent with privacy standards.
3. Audit and compliance checks to ensure alignment with UU PDP.
4. Strengthening risk governance to prevent security incidents from resulting in legal sanctions or reputational damage.

4.7.Research Evaluation

Based on Tables 4.25 and 4.26, the number of threats in the news input process was 26 with high risk and 5 with medium risk, while in the news-to-production process, 33 threats were high risk and 8 with medium risk. This is indicated by the color red for high risk and yellow for medium risk.

Next, an evaluation was conducted with PT XYZ's internal team, namely the System Procedures team, IT team, the General Manager, and managers regarding the threat findings in this CMS. Discussions were conducted based on the predetermined scope of identifying threats, vulnerabilities, and effective control mitigation based on cybersecurity standards. The results are as follows:

1. Threats have been identified in the company's CMS application with the risk scale as shown in **Tables 4.25** and **Table 4.26**.
2. Vulnerabilities have been identified in the threats shown in Tables 4.18 and 19, as well as security gaps in **Tables 4.20** and **Table 4.21**.
3. Threats and vulnerabilities at the highest level can be prioritized for mitigation control, as shown in **Tables 4.26** and **Table 4.27**, which include two mitigation controls: technical threat mitigation controls and management-level threat mitigation controls.
4. In accordance with the results of this research with the findings of threats, vulnerabilities and risks that have been identified, Stakeholders suggests that the proposed threat model framework can be implemented in the organization to reduce the impact of cyber security and increase cyber resilience in this CMS media application.
5. Further time lines can be made for the implementation of this mitigation control recommendation at PT XYZ to ensure the security of this CMS system, a critical company asset.

4.8. Research Validation

Based on the evaluation of this research, validation research is required, namely testing the research methodology used in identifying threats to PT. XYZ's CMS application by external cybersecurity experts. This validation is based on the scope form in Table 3.3, which was mutually agreed upon by the researcher and stakeholders. The purpose of this validation is to ensure that the research results are valid and can be used to identify threats, weaknesses, and mitigate risks in order to improve the security posture of PT. XYZ's CMS application to become a secure application and ensure the operational continuity of the CMS system, thus impacting PT. XYZ's business.

The following are the expert validation results:

1. The problem objective and research objectives are aligned with the company's business goals.
2. The methodology used to identify threats, vulnerabilities, and analyze risks aligns with standard threat modeling principles, starting with application decomposition, data flow diagrams, and risk evaluation.
3. **Related to Threat**, the use of STRIDE threat modeling for the security threat model in **Table 4.13**, **Table 4.14** and LINDDUN privacy threat model in **Table 4.16** and **Table 4.17** aligns with the objectives of this study, which are to improve security in CMS applications and ensure the operational continuity of CMS to support PT XYZ's business.
4. Combining multiple threat models yields more threats than using a single model alone.
5. Related to DREAD based risk analysis provides scoring based risk results, allowing for threat prioritization. This is useful for determining mitigation based on a priority scale.
6. Control mitigation, with two recommendations for engineering in **Table 4.26** and management in **Table 2.27**, provides a comprehensive and holistic solution to address this CMS problem. Therefore, this method can serve as a guideline for IT developers to incorporate threat modeling into their SDLC cycle. This will save companies time in addressing cyber threats and increase their resilience to cyberattacks.
7. According to discussions with experts, **software designs developed without considering threat modeling tend to have higher cybersecurity risks**. This is due to the absence of a systematic process for identifying potential threats, vulnerabilities, and attack scenarios early in development. Without threat modeling, developers often focus solely on functionality, resulting in security vulnerabilities only being detected after the application is deployed, when the cost and complexity of remediation have already increased significantly. Conversely, by adding threat modeling, for example using the STRIDE or LINDDUN frameworks, the design process becomes more proactive in addressing risks. These threat models enable teams to map attack entry points, classify threats based on their impact and likelihood of occurrence, and design

appropriate security controls before code is written or features are released. This approach not only reduces the number of vulnerabilities identified during the testing phase but also improves the application's overall security posture.

8. In the development of this CMS, the integration of threat modeling can anticipate common threats from the design stage, so that security risks can be significantly reduced compared to development methods that ignore this element.
9. The CMS threat model risk management framework approach applied in this thesis **can be adapted to other CMSs in general** because its principles and methods focus on identifying, analyzing, and mitigating threats originating from the system's architecture, components, and workflows, which are generally **similar across** CMS platforms. Through the use of frameworks such as STRIDE, LINDDUN for threat classification and risk management based on the OWASP Top 10 and ISO 27001 standards, this process can be applied to open source CMSs such as WordPress, Drupal, Joomla, or other custom CMSs, by adjusting the specific context, modules, and risks of each system. This approach ensures that every CMS, regardless of its programming language or implementation model, can achieve consistent security posture improvements through a systematic process of managing cybersecurity risks.

4.9. Research Discussion

Overall, the comprehensive integration of STRIDE, LINDDUN, and DREAD scoring threat modeling methodologies to develop a security risk management framework tailored to PT XYZ's Content Management System (CMS) represents a valuable step in identifying threats to the company's critical assets. This research process included systematic mapping of identified threats and vulnerabilities against corresponding security gaps in the CMS. This framework was assessed through expert interviews, case-based simulations using real-world CMS architectures, and validation of mitigation effectiveness against the top 10 OWASP threats and ISO/IEC 27001:2022 security controls. This multi-method validation ensured the relevance and applicability of the proposed model in addressing technical and organizational risk components.

The STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege) model was applied to analyze technical threats

across CMS components such as the login module, session management, APIs, and user input mechanisms. Simultaneously, LINDDUN (Linkage, Identifiability, Non-repudiation, Detection, Disclosure, Indifference, and Non-Compliance) was used to uncover privacy-related risks, particularly for CMS features that handle sensitive data from journalists and users. The combined application of STRIDE and LINDDUN enabled the identification of functional and non-functional security vulnerabilities, validating the depth and comprehensiveness of threat coverage.

To quantify and prioritize threats, a DREAD assessment was applied based on five criteria: Potential Damage, Reproducibility, Exploitability, Affected Users, and Discoverability. Each threat identified through STRIDE and LINDDUN was assessed and ranked, allowing organizations to determine high-priority risks requiring immediate control. This validation process involved cross-checking the DREAD scores with the latest CVE records relevant to the open-source CMS platform and aligning them with PT XYZ's system configuration, thus basing the assessment on real-world exploitation scenarios and empirical data.

The analysis phase mapped identified vulnerabilities such as SQL injection, session fixation, and insecure authentication to security gaps in architecture and policies. These results were further strengthened by comparing these findings to the OWASP Top 10 categories and measuring their alignment with PT XYZ's existing mitigation strategy. The gap analysis highlighted areas where existing controls were missing, misconfigured, or ineffective, which then formed the basis for the proposed risk management plan.

The results of this study indicate that the design of a structured threat modeling framework consistently helps application developers significantly improve their security posture. This structured approach guides developers in identifying, prioritizing, and mitigating potential threats from the design stage, thereby reducing the likelihood of security gaps that often arise in development processes without formal guidance. This finding aligns with studies published by (Maheshwari and Prasanna, 2016) and (Valdés-Rodríguez *et al.*, 2024), which emphasize that the integration of threat modeling into the secure software development lifecycle can transform security from a reactive to a proactive approach. Therefore, the results of this empirical evaluation support the

hypothesis that the design of a structured threat modeling framework directly contributes to improved application security.

Overall, it is confirmed that the threat modeling-based approach offers a structured and evidence-based mechanism for risk assessment and mitigation within the CMS ecosystem. The use of STRIDE and LINDDUN ensured comprehensive threat coverage, while the DREAD assessment provided measurable risk prioritization. The integration with vulnerability and security gap analysis connected theoretical modeling with practical system strengthening. This thesis presents a robust and validated model to enhance the resilience of PT XYZ's CMS to evolving cyberthreats and support compliance with ISO/IEC 27001:2022.

