

Software Design Document

Role-Based Internal Platform for Targeted Content Delivery

Version 1.0.0

Group 20
Jesus Calvo,
Justin Gip
Shane Guzman
Samuel Mekonnen

December 5, 2025
CS 3338-01 (94305)

Contents

1	Introduction	3
1.1	Purpose	3
1.2	Scope	3
1.3	Intended Audience	3
1.4	References	3
1.5	Definitions, Acronyms, and Abbreviations	4
2	System Architecture	4
2.1	Architectural Goals and Constraints	4
2.2	High Level Architecture	4
3	Component Design	5
3.1	Component Overview	5
3.2	Auth Service	5
3.2.1	Responsibilities	5
3.2.2	Interfaces	5
3.3	User and Role Management	5
3.3.1	Responsibilities	5
3.3.2	Interfaces	5
3.4	Content Management	5
3.4.1	Responsibilities	5
3.4.2	Interfaces	6
3.5	Targeting Engine	6
3.5.1	Responsibilities	6
3.5.2	Rule Model	6
3.6	Delivery Service	6
3.6.1	Responsibilities	6
3.6.2	Interfaces	6
3.7	Analytics and Reporting	6
3.7.1	Responsibilities	6
3.7.2	Interfaces	6
4	Data Design	7
4.1	Data Model Overview	7
4.2	Core Tables	7
4.2.1	User	7
4.2.2	Role	7
4.2.3	ContentItem	7
4.2.4	TargetRule	8
4.2.5	DeliveryEvent	8
5	User Interface Design	8
5.1	Screen Overview	8
5.2	Navigation Flow	8
6	Requirements Traceability	8
7	Nonfunctional Design Considerations	9
7.1	Performance	9
7.2	Security	9
7.3	Maintainability	9
8	Appendix A: Future Enhancements	9

Version Description

Version Description

Author	Date	Description	Version
Your Name	10/15/25	Initial draft of SDD created for project proposal	0.1.0
Your Name	11/05/25	Updated architecture and data design for Snapshot 2	0.2.0
Team	12/05/25	Finalized design after feedback and implementation changes	1.0.0

1 Introduction

1.1 Purpose

This Software Design Document (SDD) describes the technical level and detailed design of the **Role-Based Internal Platform for Targeted Content Delivery** (the Platform). It translates the functional and nonfunctional requirements defined in the Software Requirements Specification (SRS) into a architecture, technologies, and modules. The SDD is intended to guide developers, testers, and operations staff as they implement, test, deploy, and maintain the system.

1.2 Scope

The Platform is an internal web app used by organizations to deliver targeted announcements, training materials, and resources to employees based on their role, department, location, and other attributes. It includes:

- A responsive web user interface for end users and administrators.
- Role based authentication and authorization for all internal users.
- A content management module for creating, tagging, and scheduling content.
- A targeting rules engine that matches content to audiences.
- Delivery channels for showing content in dashboards and sending notification by email.
- Analytics and reporting for content engagement and compliance.

1.3 Intended Audience

- **Developers** use this document to understand the system architecture, module responsibilities, and interfaces.
- **Testers** use the detailed design to derive test plans, cases, and data needed to verify behavior.
- **Project managers** use the SDD to understand the scope and complexity of the solution.
- **Operations / DevOps** use the deployment view and nonfunctional design aspects.

1.4 References

- Software Requirements Specification (SRS) for Role-Based Internal Platform for Targeted Content Delivery.
- Jira space

1.5 Definitions, Acronyms, and Abbreviations

- **RBAC** Role Based Access Control.
- **CMS** Content Management System.
- **NFR** Nonfunctional Requirement.
- **SRS** Software Requirements Specification.
- **SDD** Software Design Document.

Glossary of Acronyms

Acronym	Definition
RBAC	Role Based Access Control
CMS	Content Management System
NFR	Nonfunctional Requirement
SRS	Software Requirements Specification
SDD	Software Design Document

2 System Architecture

2.1 Architectural Goals and Constraints

The system architecture is designed with the following goals:

- Enforce strong role based access control for all content and actions.
- Support flexible content targeting rules without code changes.
- Be deployable in a containerized environment (e.g., Docker).
- Be maintainable for a small team.

Constraints:

- Web based user interface built using technologies covered in class (e.g., Jira, Docker, LaTeX).
- Back end implemented as a single service to keep the project manageable.
- Uses a relational database such as PostgreSQL.

2.2 High Level Architecture

At a high level, the Platform follows a layered architecture with the following layers:

- **Presentation Layer** Web UI (end user portal and admin portal).
- **Application Layer** HTTP API, authentication, content management, targeting engine, analytics.
- **Data Layer** Relational database and persistence logic.
- **Integration Layer** Email service or notification adapter.
- Users interacting with the Web UI via browser.
- The Web UI making requests to the Application Layer (REST API).
- The Application Layer reading and writing data in the Data Layer.
- Optional outgoing calls to an email or notification service.

3 Component Design

3.1 Component Overview

The main components of the Platform are:

- **Auth Service** Handles login, logout, password reset, and enforcement of RBAC.
- **User and Role Management** Stores users, roles, permissions, and organizational attributes.
- **Content Management** Allows admins to create, edit, schedule, and retire content items.
- **Targeting Engine** Evaluates which users should see which content given a set of rules.
- **Delivery Service** Exposes content to the UI and optionally triggers outbound emails.
- **Analytics and Reporting** Aggregates view and click events and exposes metrics.

3.2 Auth Service

3.2.1 Responsibilities

- Authenticate users via email and password.
- Issue and validate session tokens or cookies.
- Attach user identity and role information to each request.

3.2.2 Interfaces

- POST /api/auth/login
- POST /api/auth/logout
- GET /api/auth/me

3.3 User and Role Management

3.3.1 Responsibilities

- Create, read, update, and deactivate user accounts.
- Maintain mapping of users to roles (e.g., Employee, Manager, HR, Admin).
- Maintain permissions per role for actions such as content creation, publishing, and reporting.

3.3.2 Interfaces

- GET /api/users
- POST /api/users
- GET /api/roles
- POST /api/roles

3.4 Content Management

3.4.1 Responsibilities

- Create and edit content items (title, body, attachments, tags).
- Schedule content start and end dates.
- Assign targeting rules to content.

3.4.2 Interfaces

- GET /api/content
- POST /api/content
- PUT /api/content/{id}
- DELETE /api/content/{id}

3.5 Targeting Engine

3.5.1 Responsibilities

- Evaluate target audiences based on user attributes (role, department, location) and rule definitions.
- Cache evaluations where possible to reduce database load.

3.5.2 Rule Model

Targeting rules are represented as a set of conditions:

- Field (e.g., role, department, location).
- Operator (equals, in list, not equals).
- Value(s) (e.g., “Manager”, “HR”, “US”).

Rules can be combined using logical AND and OR groups.

3.6 Delivery Service

3.6.1 Responsibilities

- Provide the current user with the list of active content items they should see.
- Log each view or click as an event.
- Optionally send notification emails when new content is published.

3.6.2 Interfaces

- GET /api/feed – returns targeted content for the logged in user.
- POST /api/events – records content view or click events.

3.7 Analytics and Reporting

3.7.1 Responsibilities

- Aggregate events by content item, audience, and time window.
- Expose metrics such as views, unique viewers, and completion rate for required content.

3.7.2 Interfaces

- GET /api/reports/overview
- GET /api/reports/content/{id}

4 Data Design

4.1 Data Model Overview

The Platform uses a relational database with normalized tables. Key entities include:

- User
- Role
- UserRole (join table)
- ContentItem
- TargetRule
- DeliveryEvent

4.2 Core Tables

4.2.1 User

- id (primary key)
- email (unique)
- password_hash
- first_name
- last_name
- department
- location
- is_active
- created_at

4.2.2 Role

- id (primary key)
- name (e.g., Employee, Manager, HR, Admin)
- description

4.2.3 ContentItem

- id (primary key)
- title
- body
- category
- is_mandatory (boolean)
- start_date
- end_date
- created_by_user_id (foreign key to User)

4.2.4 TargetRule

- id (primary key)
- content_id (foreign key to ContentItem)
- field (e.g., “role”, “department”, “location”)
- operator (e.g., “EQUALS”, “IN”)
- value

4.2.5 DeliveryEvent

- id (primary key)
- content_id (foreign key)
- user_id (foreign key)
- event_type (view, click, dismiss)
- occurred_at (timestamp)

5 User Interface Design

5.1 Screen Overview

The main screens are:

- **Login Screen** Authenticates the user into the Platform.
- **End User Dashboard** Shows a personalized feed of targeted content items, including filters and search.
- **Content Detail Page** Displays full content, attachments, and required acknowledgement actions.
- **Admin Content List** Lists all content items with status, author, and filters.
- **Content Editor** Form for creating and editing content and targeting rules.
- **Reports Dashboard** Displays analytics charts and tables.

5.2 Navigation Flow

Typical navigation flow:

1. User navigates to the login page and enters credentials.
2. After successful authentication, the Platform redirects to the appropriate dashboard:
 - Regular employees see the End User Dashboard.
 - Admins see the Admin Content List or Reports Dashboard.
3. From the dashboard, users can click into a content item, acknowledge required content, or mark optional content as read.
4. Admin users can navigate to content creation, targeting configuration, and reporting views.

6 Requirements Traceability

Table 3 demonstrates the requirements and its design

Requirements Traceability

Requirement ID	Requirement (from SRS)	Design Element
FR-01	The system shall authenticate users using organization credentials.	Auth Service, Login Screen
FR-02	The system shall allow admins to create and publish content items.	Content Management, Admin Content List, Content Editor
FR-03	The system shall show each user only the content targeted to them.	Targeting Engine, Delivery Service, End User Dashboard
FR-04	The system shall track when users view or acknowledge content.	DeliveryEvent table, Delivery Service, Reports Dashboard

7 Nonfunctional Design Considerations

7.1 Performance

- Use indexed columns on foreign keys and frequently filtered fields.
- Cache the list of targeted content for a user within a short time window to reduce rule evaluations.

7.2 Security

- All authenticated endpoints require a valid session or token.
- Sensitive data, such as password hashes, is never logged or exposed through APIs.
- RBAC checks are enforced in a single middleware or service to avoid duplication.

7.3 Maintainability

- Follow consistent coding standards and naming conventions.
- Organize the code base by feature or layer.
- Write unit tests for core business logic, especially the targeting engine.

8 Appendix A: Future Enhancements

Potential future enhancements include:

- Mobile push notifications.
- A self service rule builder with a graphical interface.
- Integration with learning management systems for training content.