National Engineering Services Pakistan (NESPAK)

Software Requirements Specification

Nespak Digital Learning & Representation Platform

By Asra Bukhari
Software Engineer - Intern

1. Introduction

1.1 Purpose

This document defines the functional and non-functional requirements for Nespak's Internal Digital Learning & Knowledge Platform. The platform will centralize employee training, capacity-building resources, organizational preferences, and project-related documents. It will also provide mechanisms for secure access, progress tracking, and structured feedback collection to support continuous improvement.

1.2 Scope

The platform will feature a dashboard with 4 core sections:

- Trainings & Development / Capacity Building
- Nespak Representation
- Nespak Preferences
- Project-Related Documents

Core functionalities:

- Admins can upload training materials (YouTube video links, slides, metadata).
- Users can securely view, search, and track content.
- User authentication includes email verification to ensure secure registration.
- Content will be enhanced with tags and categorized under sections.
- User activity will be logged for analytics (views, progress).
- A structured feedback system will capture user ratings and suggestions.
- A request module exists but remains hidden as per current supervisory directive.

1.3 Intended Audience

- Nespak management & decision-makers
- Software development team
- Internal content administrators (admins)
- End-users (Nespak employees)

---

2. Overall Description

2.1 Product Perspective

This is a new standalone web application integrated within Nespak's internal ecosystem. It leverages Microsoft SQL Server for structured storage, Node.js (Express.js) for backend APIs, and a React-based frontend for user interaction.

2.2 User Classes

- **Admin:** Upload/manage content (YouTube video + PDF slides), assign tags, moderate feedback, and review analytics.
- **Viewer:** Browse, search, and view content with progress tracking; submit feedback.

2.3 Constraints

- Accessible only within Nespak's internal secure network.
- Videos must be embedded using YouTube links (local video uploads not supported).
- PDF slides via google drive (link upload).
- Requests feature exists but is disabled in production as per supervisor directive.

2.4 Functional Requirements

ID	Requirement
FR-1	The system shall display a dashboard with 4 sections.
FR-2	Admin shall upload content including YouTube link, title, description, speaker, section, and difficulty level.
FR-3	Admin shall upload supporting slides in PDF format.
FR-4	Users shall view and download slides, and stream videos via embedded YouTube player.
FR-5	Content shall be categorized and filterable by section, tag, speaker, or difficulty level.
FR-6	Admin Panel shall include forms for content upload, editing, and soft deletion.
FR-7	The system shall enforce login-based access (Admin / Viewer roles).
FR-8	The system shall include email verification during signup (via PendingVerifications table).
FR-9	The system shall track user activity (views, timestamps, progress %).
FR-10	The system shall allow search across content by title, tags, section, or speaker.
FR-11	The system shall support tagging with multiple tags per content item.
FR-12	The system shall allow users to submit feedback (message, rating, optional topic).
FR-13	Admins shall be able to review and mark feedback as "read/old".

ID	Requirement
FR-14	The system shall provide analytics dashboards for admins (uploads, views, progress, section stats).
FR-15 (Hidden)	The system shall include a request module (submit, approve/reject requests) but it shall remain disabled in production until activated.

2.5 Non-Functional Requirements

ID	Requirement
NFR-1	The platform shall have a responsive UI (desktop, tablet, mobile).
NFR-2	The embedded video player shall support seeking, pause/resume, and fullscreen.
NFR-3	Uploads (slides) shall be validated for file type (PDF) and size limits; video input shall be validated as a YouTube URL.
NFR-4	Platform shall load content within ≤ 2 seconds on standard enterprise internet.
NFR-5	The system shall be modular and scalable for future features (comments, assignments, advanced analytics).
NFR-6	User data (auth, activity, feedback) shall be stored securely with referential integrity.
NFR-7	Metadata shall be indexed for fast search and filtering.
NFR-8	Tagging system shall support multiple tags per content item.
NFR-9	Authentication shall use JWT with expiration to prevent unauthorized access.
NFR-10	Sensitive data (passwords) shall be hashed using industry standards (bcrypt).
NFR-11	The system shall restrict file uploads to prevent malicious content.
NFR-12	Role-based route protection shall ensure viewers cannot access admin-only features.

---

3. Database Design

The backend is structured around a normalized relational database using Microsoft SQL Server. The schema supports the core functionality of the NESPAK LMS, including user management, verification, training content organization, tagging, feedback collection, request handling, and activity tracking.

3.1 Overview of Schema Components

- **Users**
  Stores registered employees and administrators.
  Includes fields for name, email (unique), password hash, role (admin/viewer), account verification status, and creation date.
  This table enforces authentication and role-based access control.

- **PendingVerifications**
  Temporary table used during user registration.
  Holds user details and verification codes until email verification is completed.

- **Sections**
  Represents the four core dashboard categories:

  - Trainings & Development
  - Nespak Representation
  - Nespak Preferences
  - Project-Related Documents

  Ensures structured organization and scalable classification of uploaded materials.

- **Content**
  Central table for all uploaded training/project materials.
  Each record contains:

  - Title, description, speaker name
  - Media links (video URL, slide URL)
  - Difficulty level (beginner, intermediate, advanced)
  - Section reference
  - Uploader reference (admin user)
  - Upload timestamp
  - Soft deletion flag (is_deleted) for recoverability

  Enforces integrity via foreign keys to Sections and Users.

- **Tags**
  Stores reusable tag names for categorizing content (e.g., department, topic, project type).

- **ContentTags**
  Many-to-many relationship table linking Content and Tags, allowing flexible search and categorization.

- **Views**
  Tracks employee interaction with training content.
  Stores viewer, viewed content, timestamp, and progress percentage (0–100).
  Enables analytics on learning progress and employee engagement.

- **Requests**
  Handles access or info requests from external users
  Each request includes: requester details, category, and workflow status (pending, granted, rejected).

- **Feedback**
  Collects user feedback on content or platform experience.
  Each feedback includes: message, optional topic, rating (1–5), creation timestamp, status (new/old), and optional review timestamp.
  Linked to Users to identify the feedback provider.

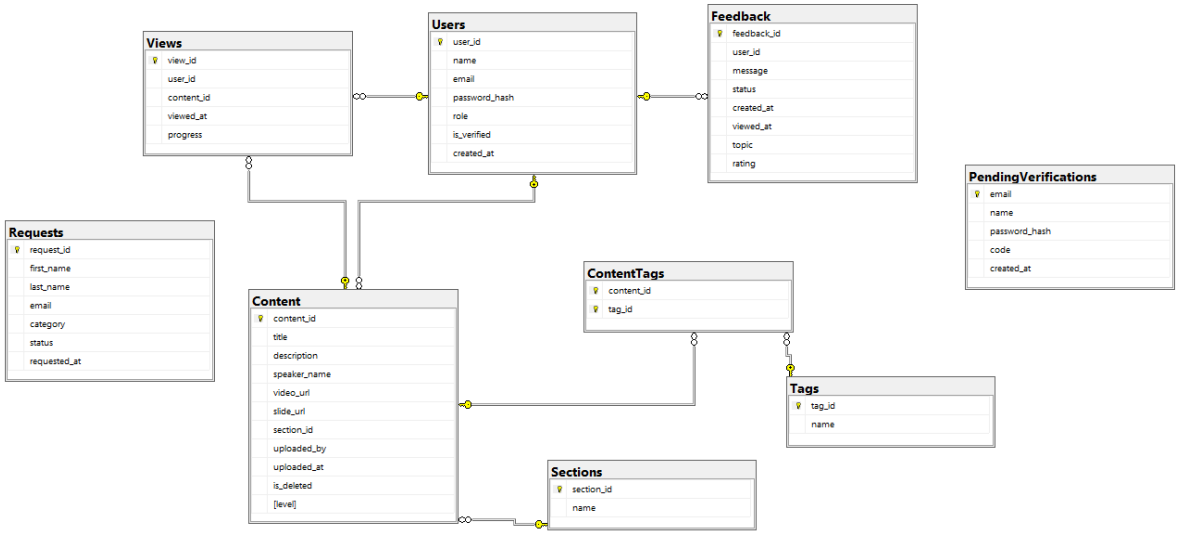
3.2 Additional Details

- **Referential Integrity:** All relationships are enforced through foreign key constraints. Cascading deletes are used in ContentTags and Feedback to ensure dependent records are automatically removed**.**

- **Indexes for Optimization:**
  - idx_content_title → speeds up content title search
  - idx_content_speaker → optimizes queries by speaker
  - idx_tags_name → efficient tag lookups
  - idx_views_user → improves analytics by user progress
  - idx_content_is_deleted → filters active vs deleted content quickly
  - idx_feedback_status and idx_feedback_user → optimizes feedback review workflows
- **Scalability:**
  The schema is modular, allowing easy integration of new features such as:
  - Additional content types (e.g., PDFs, assignments)
  - Feedback workflows (admin replies, escalation)
  - Advanced analytics (completion tracking, performance dashboards)

3.3 Database Diagram

A visual ER diagram can be inserted here showing the relationships between Users, Content, Sections, Tags, Feedback, Requests, and Views.



---

4. Backend Overview

The backend provides a secure, scalable REST API to support user authentication, content management, feedback collection, analytics, and capacity-building features for NESPAK employees. It is built on Node.js with Microsoft SQL Server, ensuring enterprise-level robustness and integration with existing infrastructure.

Core functionalities include:

- User login, registration, and verification
- Role-based access (admin/viewer)
- Content upload & management (videos via YouTube links, PDF slides)
- Tagging and classification of content
- Progress tracking (views & completion percentage)
- Feedback system (with status tracking for admins)
- Dashboard analytics (user activity, uploads, completions, etc.)
- Hidden (future) request management module (employee requests for access/resources)

4.1 Backend Tech Stack

Area	Technology Used	Reason
Language	Node.js (Express.js)	Lightweight, asynchronous, ideal for building REST APIs.
Database	Microsoft SQL Server	Enterprise-grade RDBMS, reliable, secure, scalable.
Database Access	mssql (SQL driver)	Direct SQL queries, avoids ORM overhead, better query optimization/control.
Authentication	JWT (JSON Web Tokens)	Stateless, secure, widely adopted for session management.
File Storage	Local/Drive	For storing uploaded PDF slides.
Video Storage	YouTube (embedded links)	Prevents large video storage costs, reliable streaming.
API Testing	Postman	Convenient for manual endpoint testing.
Deployment	Nespak's Cloud Server	Internal secure hosting, fits organizational needs.

4.2 Authentication & Roles

- JWT-Based Authentication with middleware enforcement.
- Email Verification during registration (verification codes stored in PendingVerifications table).
- **Roles:**
  Admin: Upload/manage content, assign tags, view analytics, moderate feedback
  Viewer: Watch videos, download slides, track personal progress

4.3 Backend Modules

Module	Description
Auth Module	Login, register, verification, password hashing, JWT-based sessions
User Module	Fetch user profile, user-related activity (views, progress, uploads)

Module	Description
Content Module	Upload/manage YouTube video links & PDF slides, soft-delete, sectioned data
Tag Module	Add/search/remove tags, map tags to content (many-to-many)
View Module	Track views, progress percentage, completion status
Dashboard Module	Analytics for admins (views, completions, uploads, section breakdowns)
Feedback Module	Submit feedback, track status (new/old), rating (1–5), admin review
Request Module (Hidden)	Employee requests (pending/granted/rejected) – currently disabled as per supervisor directive

4.4 RESTful API Endpoints

Auth

Method	Endpoint	Description
POST	/api/auth/register	Register new user + send verification
POST	/api/auth/verify	Verify email with code
POST	/api/auth/login	Login with email/password, issue JWT

Users

Method	Endpoint	Description
GET	/api/users/me	Get logged-in user profile
GET	/api/users/:id/views	Get content viewed by a user

Content

Method	Endpoint	Description
GET	/api/content	Get all content (filter by tags, section, level)
GET	/api/content/:id	Get content details
POST	/api/content (admin)	Upload new content (video/slide)
PUT	/api/content/:id (admin)	Edit content details
DELETE	/api/content/:id (admin)	Soft delete content

Tags

Method	Endpoint	Description
GET	/api/tags	Get all tags
POST	/api/tags (admin)	Create new tag
DELETE	/api/tags/:id (admin)	Delete tag
POST	/api/content/:id/tags	Assign tags to content

Views

Method	Endpoint	Description
POST	/api/views	Log user view + progress
GET	/api/content/:id/views	Get all views for a content item
GET	/api/analytics/user/:id	User-specific progress analytics

Feedback

Method	Endpoint	Description
POST	/api/feedback	Submit feedback with optional rating
GET	/api/feedback (admin)	View feedback by status

Method	Endpoint	Description
PUT	/api/feedback/:id (admin)	Mark feedback as read/old

Dashboard

Method	Endpoint	Description
GET	/api/dashboard (admin)	Overall analytics (uploads, views, completions, section stats, user stats)

Requests (Hidden)

This module is implemented but currently disabled as per supervisor directive.
Endpoints (not exposed in production yet):

POST /api/requests → Submit employee request

GET /api/requests → View requests (admin)

PUT /api/requests/:id → Approve/reject requests

4.5 Folder Structure

```
backend/
├── controllers/      # Business logic (auth, content, feedback, dashboard)
├── routes/           # Express routes mapped to controllers
├── middleware/       # JWT auth, role validation, error handling
├── utils/            # Helper functions
├── uploads/          # PDF storage (for slides)
├── db.js             # MSSQL connection pool
├── server.js         # Main entry point
└── .env              # Environment variables
```

4.6 Security Considerations

- JWT expiration implemented (1d), refresh flow can be added.
- Password hashing using bcrypt.
- Role-based route protection via middleware.
- Soft delete for content to prevent accidental loss.
- Input validation (title length, email uniqueness, file type).
- Restricted file upload types (PDF only for slides, YouTube embedding for videos).

5. Frontend Overview

The frontend is a modern, responsive web application built with React and Tailwind CSS. It provides a professional, user-friendly experience for Nespak employees while adhering to Nespak's branding. The interface is role-based, ensuring Admins and Viewers see only the functionality relevant to them.

5.1 Frontend Objectives

- Provide a clean and intuitive dashboard with access to the 4 core sections.
- Deliver separate role-based experiences for Admins (content management, user management, feedback) and Viewers (content browsing and learning).
- Support video-based learning with progress tracking and completion marking.
- Ensure smooth slide viewing/downloading alongside video content.
- Maintain responsiveness and accessibility across desktop, tablet, and mobile.

5.2 Frontend Tech Stack

Area	Technology Used	Reason
Language	TypeScript (React)	Type safety and scalability.
Framework	React.js	Component-based, fast, scalable.
UI Library	Tailwind CSS + shadcn/ui	Utility-first styling and prebuilt UI components for consistent branding.
Routing	React Router DOM	Client-side routing for multiple pages (Dashboard, Content, etc.).
State Management	React Hooks + Context	Lightweight state management for auth and role-based access.
Video Player	react-youtube	Full YouTube embedding with playback control and progress tracking.
File Handling	Axios	Secure API calls for CRUD, uploads, and downloads.
Deployment	Nespak Cloud Hosting	Secure internal deployment within enterprise environment.

5.3 Frontend Folder Structure

```
frontend/
│
├── public/                  # Static assets
│   ├── upload/              # Images (Nespak building, etc.)
│   └── favicon.ico          # App icon
│
├── src/
│   ├── assets/              # Logos, static images
│   │   └── nespak-logo.png
│   │
│   ├── components/          # Reusable UI and layout components
│   │   ├── branding/        # Branding elements
│   │   │   └── Header.tsx
│   │   ├── layout/          # Layout wrappers
│   │   │   └── DashboardLayout.tsx
│   │   ├── sections/        # Section blocks for landing page
│   │   └── ui/              # UI components (Button, Card, Input, Badge, etc.)
│   │
│   ├── hooks/               # Custom React hooks
│   │   ├── use-mobile.tsx
│   │   └── use-toast.ts
│   │
│   ├── lib/                 # Utility functions
│   │
│   ├── pages/               # Page-level components
│   │   ├── Index.tsx        # Landing page
│   │   ├── Login.tsx        # Login screen
│   │   ├── Signup.tsx       # Signup screen
│   │   ├── Dashboard.tsx    # User dashboard
│   │   ├── TrainingDevelopment.tsx
│   │   ├── NespakPreferences.tsx
│   │   ├── NespakRepresentation.tsx
│   │   ├── ProjectDocuments.tsx
│   │   ├── ContentPage.tsx  # Detailed content view (video, slides, progress)
│   │   └── AdminPanel.tsx   # Admin-only controls (content, users, feedback)
│   │
│   ├── main.tsx             # App entry point
│   └── App.tsx              # Root components
└── .env                     # API Base URL
```

5.4 Frontend Features

Feature	Description
Responsive UI	Layout adapts seamlessly to mobile, tablet, and desktop.
Role-Based Dashboard	Admins can upload, update, delete content, manage users, and review feedback. Viewers can browse, watch, and download slides.
Content Viewer	Videos embedded via YouTube with real-time progress tracking, resume capability, and completion marking.
Slides Viewer	Downloadable PDFs linked to each training module.
Search & Filters	Content can be filtered by section, speaker, tags, or title.
Tagging System	Tag badges displayed with content for quick classification.
User Progress Tracking	Dynamic progress bar and "Completed" status for each viewer.
Admin Actions	Secure modals for updating or deleting content (restricted to Admins).
Toast Notifications	User-friendly success/error alerts for interactions (update, delete, etc.).
Branding	Consistent use of Nespak logo, favicon, and images for corporate identity.

5.5 Security Considerations

- Client-side route protection based on JWT role validation.
- Admin-only features hidden from viewers.
- Validation of all uploaded and embedded URLs to prevent malicious content.
- Secure API communication with HTTPS.
- Session session storage only for user ID and JWT (cleared on logout).
- Protected AdminPanel routes ensure unauthorized users cannot bypass role restrictions.
