**Software Requirement Specifications**

**Ordered Network Analysis Model for Sequenced Learning Activities**



**Submitted by:** Umyma Mohsin ( F22BINFT1M01278 )

**Submitted to:** Sir Mustafa Hameed

**Department of Information Technology**

**Faculty of Computing**

**The Islamia University of Bahawalpur**

**Meeting Details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr No** | **Details** | **Date** | **Supervisor Signature** |
| SR01 | Discussed project objectives, scope, and system requirements. | 26/11/25 |  |
| SR02 | Reviewed functional requirements, use cases, and workflow processes. | 3/12/25 |  |
|  |  |  |  |
|  |  |  |  |

**Summary**

The ONA System is a web-based educational planning tool designed to help instructors and curriculum designers organize learning activities in a structured, dependency-based sequence. The system allows bulk input of activities via CSV datasets, defines prerequisite relationships, detects structural issues, generates interactive network graphs, and produces an optimized learning sequence. Analytics and export options are included to support curriculum planning.

**Table of Contents**

[**1.** **Introduction** 4](#_Toc215843866)

[**1.1.** **Purpose** 4](#_Toc215843867)

[**1.2.** **Scope** 5](#_Toc215843868)

[In-Scope Functionalities 5](#_Toc215843869)

[Out of Scope 5](#_Toc215843870)

[**1.3.** **Product Perspective** 5](#_Toc215843871)

[**1.4.** **User Characteristics** 5](#_Toc215843872)

[**1.5.** **Similar apps and systems/Literature Review** 6](#_Toc215843873)

[Gap Identified 6](#_Toc215843874)

[**1.6.** **Proposed Technologies** 6](#_Toc215843875)

[**2.** **Requirements** 6](#_Toc215843876)

[**2.1.** **Functional Requirements** 7](#_Toc215843877)

[**2.2.** **Non-Functional Requirements** 8](#_Toc215843878)

[**8.** **Use Cases and Flow of Processes** 8](#_Toc215843879)

[**8.1.** **Use Case 1: Upload Learning Activities** 9](#_Toc215843880)

[**8.2.** **Use Case 2: Manual Activity Entry** 11](#_Toc215843881)

[**8.3.** **Use Case 3: Generating Optimized Network Graph** 14](#_Toc215843882)

[**8.4.** **Use Case 4: Generating Optimized Learning Sequence** 15](#_Toc215843883)

[**8.5.** **Use Case 5: Analytics and Error Detection** 17](#_Toc215843884)

[**8.6.** **Use Case 6: Export Data** 18](#_Toc215843885)

[**9.** **References** 19](#_Toc215843886)

1. **Introduction**

The ONA System is developed as a web-based tool to assist academic institutions in sequencing learning activities. Many educators face challenges in course design due to improper sequencing or missing prerequisites. This system enables users to upload datasets of learning activities, define dependencies, and automatically generate visual graphs and optimized sequences.

* 1. **Purpose**

The purpose of this project is to:

* Provide educators with a tool to **organize learning activities in a correct order** based on prerequisite relationships.
* Automatically generate a **visual dependency graph** that displays how learning tasks are interconnected.
* Produce an optimized **learning sequence** that follows logical progression from foundational topics to advanced topics.
* Identify issues such as missing prerequisites, loops, bottlenecks, or poorly-structured content.
* Support academic planning, instructional design, and curriculum development through data-driven analysis.

The project’s core objective is to improve the quality of course design by ensuring that learning activities are **well-structured, logically sequenced, and easy to visualize**.

* 1. **Scope**

**In-Scope Functionalities**

This project will include:

 Bulk activity input via CSV files (main method).

 Optional manual entry of individual activities.

Automatic network graph generation and sequence optimization.

 Analytics dashboard for cycles, bottlenecks, and orphan nodes.

 Data export in PDF, PNG, or CSV formats.

**Out of Scope Functionalities**

The project **will not** include:

* User authentication/login system.
* Full Learning Management System (LMS) features.
* AI-based recommendation engines.
* Collaboration features (multiple users editing at the same time).
* Content delivery or e-learning modules (videos, quizzes, exams).
  1. **Product Perspective**

The module functions as an **interactive planning and visualization tool**, enabling users to input activities and immediately see the structure of their course. It does not replace an LMS; rather, it complements existing educational platforms by offering a clear and intelligent sequencing engine.

* 1. **User Characteristics**

**Primary Users:**

* Instructors / Curriculum Designers: input activities, define prerequisites, generate sequences.

**Skill Requirements:**

* Basic computer literacy; no programming knowledge required
  1. **Similar apps and systems/Literature Review**

The following use ONAs in their systems:

 **Learning Management Systems (e.g., Moodle, Canvas)**

* Strengths: Provide course delivery, assignments, and basic instructional tools.
* Limitations: Do not support dependency-based sequencing of learning activities.

 **Concept Mapping Tools (e.g., CmapTools, MindMeister)**

* Strengths: Enable visual brainstorming and idea mapping.
* Limitations: Not designed for structured prerequisite-based course sequencing.
  1. **Proposed Technologies**

The following technologies will be used in the development of the system:

**Frontend:** React.js, Tailwind CSS, Cytoscape.js, JavaScript ES6  
**Backend:** Python Flask, Flask-RESTful, NetworkX  
**Database:** SQLite (or MySQL)  
**Data Handling:** Pandas  
**Visualization / Analytics:** Cytoscape.js, Chart.js  
**Tools:** VS Code, GitHub, Postman, Figma/Draw.ios

1. **Requirements**
   1. **Functional Requirements**

**FR001 – Dataset Input (CSV Upload)**

* **Name:** FR001
* **Purpose:** To allow bulk import of learning activities via files (CSV or other).
* **User(s):** Instructor, Curriculum Designer
* **Input:** CSV file containing Activity Name, Description, Prerequisites (comma-separated), Category/Type
* **Output:** Activities are stored in the system and added to the network graph.

**FR002 – Manual Activity Entry**

* **Name:** FR002
* **Purpose:** To allow addition of a single learning activity manually.
* **User(s):** Instructor, Curriculum Designer
* **Input:** Activity Name, Description, Prerequisites, Category/Type
* **Output:** Activity is saved in the system and added to the network graph.

**FR003 – View Learning Activities**

* **Name:** FR003
* **Purpose:** To display all added learning activities in a list.
* **User(s):** Instructor, Curriculum Designer
* **Input:** None
* **Output:** Full list of activities with key details.

**FR004 – Generate Network Graph**

* **Name:** FR004
* **Purpose:** To create a visual network graph showing activities as nodes and prerequisites as edges.
* **User(s):** Instructor, Curriculum Designer
* **Input:** List of learning activities and their prerequisite relationships
* **Output:** Interactive network graph with zoom, pan, and hover details.

**FR005 – Generate Optimized Learning Sequence**

* **Name:** FR005
* **Purpose:** To produce a recommended sequence of learning activities based on prerequisites.
* **User(s):** Instructor, Curriculum Designer
* **Input:** Activities and dependencies
* **Output:** Ordered list of activities from foundational to advanced topics.

**FR006 – Analytics and Error Detection**

* **Name:** FR006
* **Purpose:** To analyze the network and detect issues such as cycles, orphan nodes, and bottlenecks.
* **User(s):** Instructor, Curriculum Designer
* **Input:** Complete network of activities
* **Output:** Analytics dashboard with warnings, statistics, and summaries.

**FR007 – Export Data**

* **Name:** FR007
* **Purpose:** To allow users to export the network graph or learning sequence.
* **User(s):** Instructor, Curriculum Designer
* **Input:** Selection of export format (PDF, PNG, CSV)
* **Output:** Downloadable file of the chosen data
  1. **Non-Functional Requirements**

 **Usability:** The system must be intuitive and easy to use for users with basic computer literacy.

 **Performance:** Generating the network graph and sequence should occur within 5–8 seconds for datasets under 500 activities.

 **Reliability:** Validation ensures consistent inputs.

 **Maintainability:** Code should be modular and well-documented to allow future updates or feature extensions.

 **Portability:** The system must be accessible on modern web browsers

1. **Use Cases and Flow of Processes**

The system-level use cases are directly based on the functional requirements defined in Section 2 and describe how actors interact with the system to achieve project goals.

**Actors:**

* **Instructor / Curriculum Designer:** Main user adding and editing learning activities.

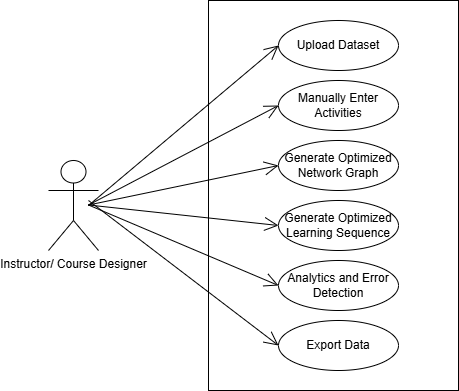
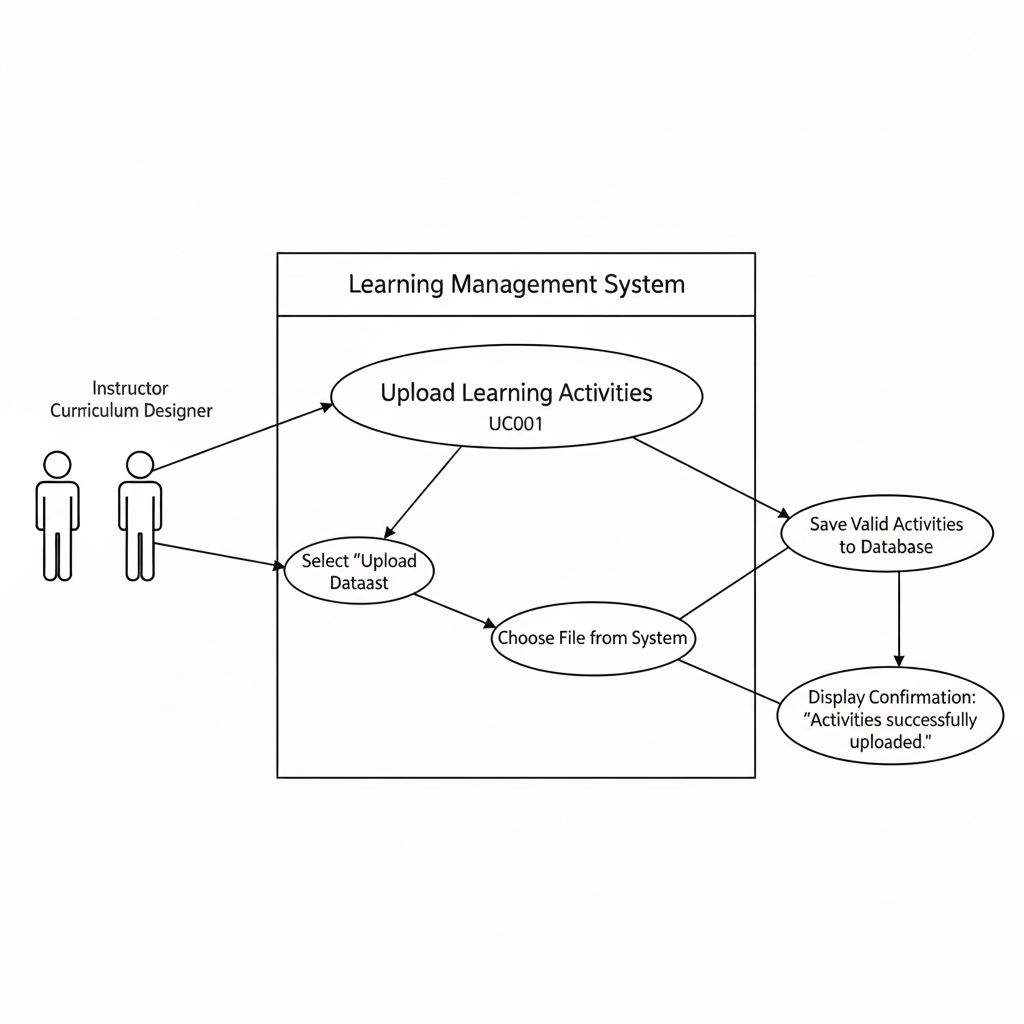
****

Figure 1: System Level Use Case Diagram

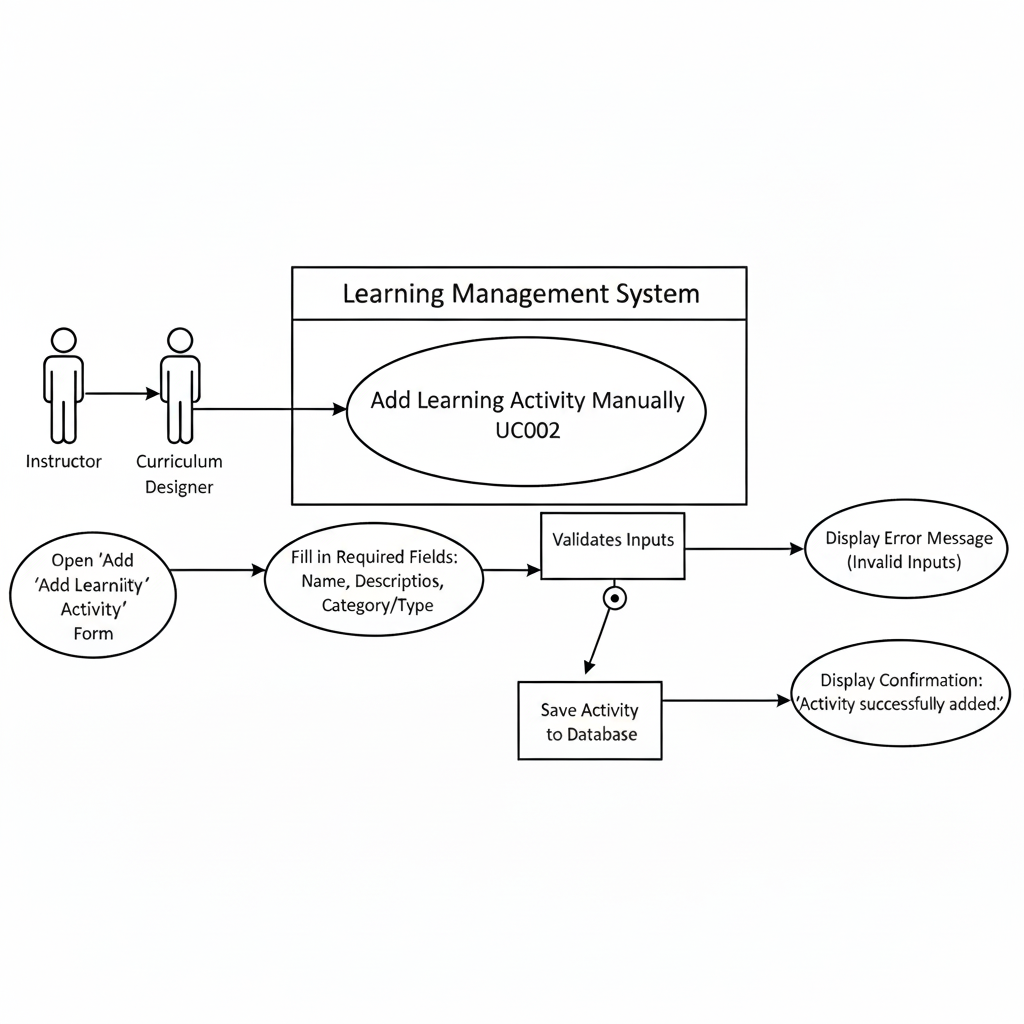
* 1. **Use Case 1: Upload Learning Activities**

|  |  |
| --- | --- |
| **ID** | UC001 |
| **Name** | Upload Learning Activities |
| **Description** | This use case describes the process of uploading a dataset containing multiple learning activities into the system. |
| **Requirement(s)** | FR001 |
| **Actor(s)** | Instructor, Curriculum Designer |
| **Precondition** | Actor has a dataset with properly formatted learning activities |
| **Postcondition** | All activities from the dataset are added to the system and become part of the network graph |
| **Basic Flow** | 1. Actor selects the “Upload Dataset” option. 2. Actor chooses the file from their system. 3. System saves all valid activities to the database. 4. System displays a confirmation message: “Activities successfully uploaded.” |

****

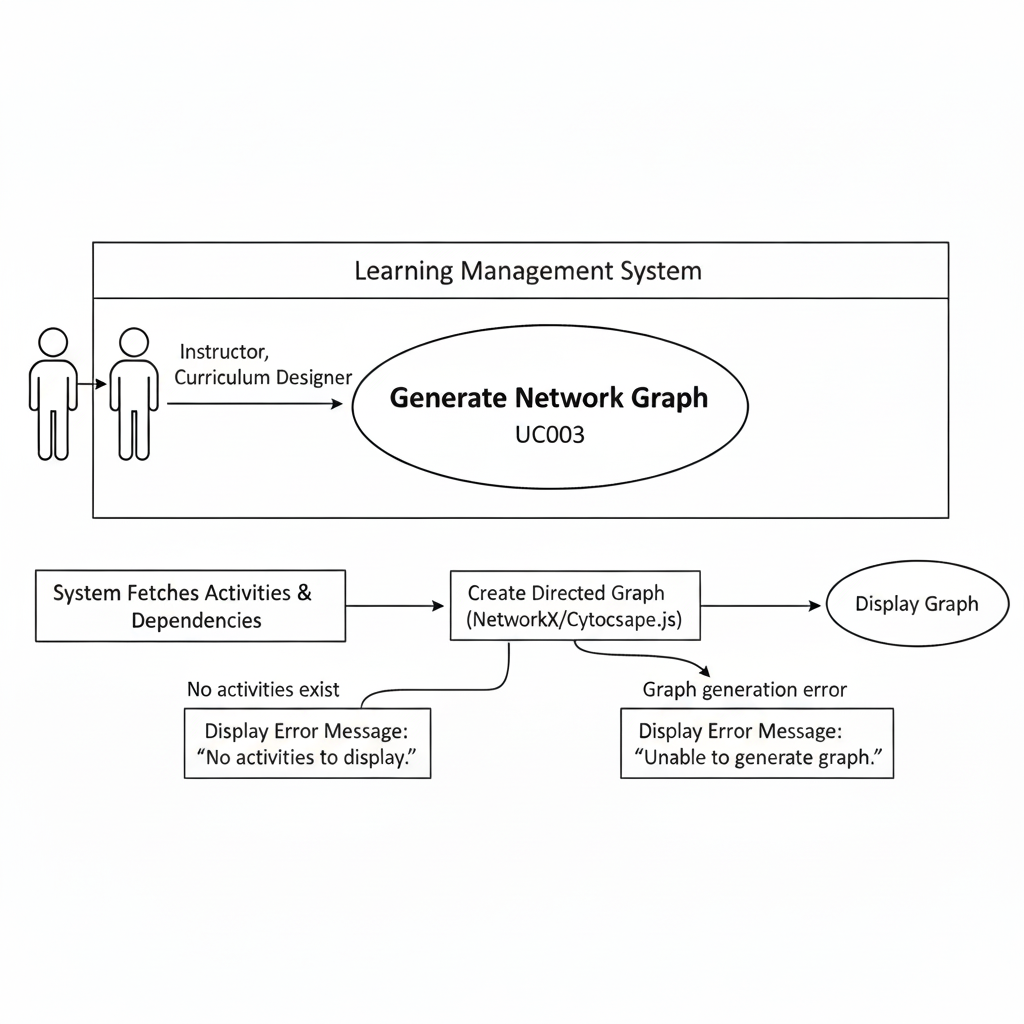
* 1. **Use Case 2: Manual Activity Entry**

|  |  |
| --- | --- |
| **ID** | UC002 |
| **Name** | Add Learning Activity Manually |
| **Description** | Allows the user to add a single learning activity directly. |
| **Requirement(s)** | FR002 |
| **Actor(s)** | Instructor, Curriculum Designer |
| **Precondition** | Actor is ready to add a new activity. |
| **Postcondition** | Activity is added to the system and included in the network graph |
| **Basic Flow** |  Actor opens the “Add Learning Activity” form.   Actor fills in required fields: Activity Name, Description, Prerequisites, Category/Type.   System validates inputs   On validation success, system saves the activity.   System displays confirmation: “Activity successfully added.” |

****

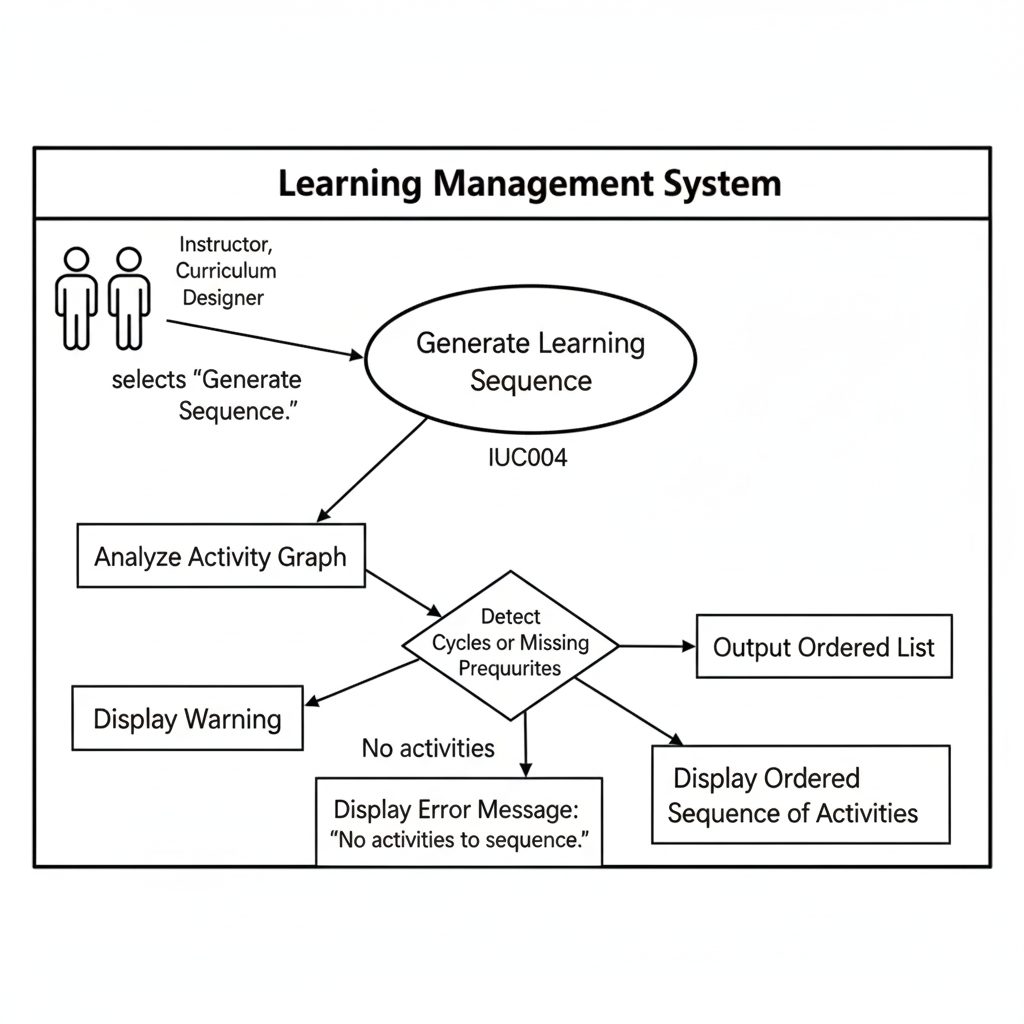
* 1. **Use Case 3: Generating Optimized Network Graph**

|  |  |
| --- | --- |
| **ID** | UC003 |
| **Name** | Generate Network Graph |
| **Description** | Generates an interactive visual graph of all learning activities and their prerequisites. |
| **Requirement(s)** | FR004 |
| **Actor(s)** | Instructor, Curriculum Designer |
| **Precondition** | At least one learning activity exists in the system. |
| **Postcondition** | Network graph is displayed. |
| **Basic Flow** | 1. Actor clicks “Generate Graph.” 2. System fetches all activities and dependencies. 3. System uses NetworkX/Cytoscape.js to create a directed graph. 4. System displays graph.   **Exceptions:**   * No activities exist → “No activities to display.” * Graph generation error → “Unable to generate graph.” |

****

* 1. **Use Case 4: Generating Optimized Learning Sequence**

|  |  |
| --- | --- |
| **ID** | UC004 |
| **Name** | Generate Learning Sequence |
| **Description** | Produces a recommended sequence of activities based on prerequisites. |
| **Requirement(s)** | FR005 |
| **Actor(s)** | Instructor, Curriculum Designer |
| **Precondition** | Activities with defined prerequisites exist |
| **Postcondition** | Ordered sequence of activities is displayed |
| **Basic Flow** | 1. Actor selects “Generate Sequence.” 2. System analyzes the activity graph. 3. System detects cycles or missing prerequisites. 4. System outputs an ordered list from foundational to advanced topics.   **Exceptions:**   * Cyclic dependencies → system displays warning * No activities → “No activities to sequence.” |

****

* 1. **Use Case 5: Analytics and Error Detection**

|  |  |
| --- | --- |
| **ID** | UC005 |
| **Name** | Analytics and Error Detection |
| **Description** | Provides system-level analysis and highlights errors like orphan nodes, bottlenecks, or cycles |
| **Requirement(s)** | FR006 |
| **Actor(s)** | Instructor, Curriculum Designer |
| **Precondition** | Activities and dependencies exist |
| **Postcondition** | Dashboard shows statistics, warnings, and graphs |
| **Basic Flow** | 1. Actor clicks **“View Analytics”** button. 2. System scans all activities and dependencies. 3. System calculates statistics 4. System displays warnings.   **Exceptions:**   * No activities → system displays “No analytics available.” * Analysis fails → system shows “Unable to generate analytics.” |

* 1. **Use Case 6: Export Data**

|  |  |
| --- | --- |
| **ID** | UC006 |
| **Name** | Export Data |
| **Description** | Allows users to export network graph or sequence as PDF, PNG, or CSV. |
| **Requirement(s)** | FR007 |
| **Actor(s)** | Instructor, Curriculum Designer |
| **Precondition** | Activities and sequences exist |
| **Postcondition** | Selected data is exported successfully. |
| **Basic Flow** | 1. Actor selects **export type** (PDF, PNG, CSV). 2. Actor clicks **“Export”** button. 3. System fetches the relevant data (network graph or sequence). 4. System generates the export file. 5. System prompts the actor to **download the file**.   **Alternative Flow:**   * Actor changes export type before exporting → system updates output format.   **Exceptions:**   * No data available → system shows “Nothing to export.” * Export fails due to system error → system shows “Export failed, try again.” |

1. **References**

[1] J. Wang and F. Zhang, “Design of a learning path recommendation system based on a knowledge graph,” International Journal of ICT in Education, vol. 19, no. 1, 2023. Available: <https://www.sciencedirect.com>

[2] NetworkX Developers, “NetworkX — Python library for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks,” 2025. Available: <https://networkx.org>