Guide on the Side Project

Project Members:

Kadesh Yahaya

Brian Ezeokeke

Habib Adelagun

Chinwike Martin Okonkwo

Luis Mondragon Verduzco

Bruno Oshiokpekhai

Charlottetown, Prince Edward Island December 15, 2025

Table Of Contents

Table of Contents

Guide on the Side Project ***1***

Research & Setup **35**
Identify Pressbooks Integration Mechanisms 35
Determine Two-Pane Tutorial Presentation Model 35
Identify Required Authoring Capabilities 35
Evaluate Rich-Text Editing Options 36
Define Authentication and Role Requirements 36
Identify Analytics Requirements for Dashboard 36
Establish a Development Workflow 36

Tutorial Data Representation **37**
Define Tutorial Representation 37
Define Slide Representation 37
Define Question Representation 37
Define Branching Representation 37

Tutorial Creation & Management Services **38**
Create the Tutorial Service 38
Retrieve the Tutorial Service 38
Update the Tutorial Service 38
List the Tutorials Service 38

Authoring Interface **39**
Display the Author Dashboard 39
Create a Tutorial UI Flow 39
Allow Authors to Manage Slides 39
Allow Authors Edit Slide Content 40
Allow the Authors to Manage Questions 40
Allow authors Configure Branching Rules 40
Allow Authors Preview Tutorial 40
Undo / Redo Support 41
Autosave Editing State 41

Tutorial Playback (Student Experience) **41**
Render Two-Pane Layout 41
Allow students Navigate Slides 41
Display Questions & Record Response 42
Execute Branching Logic 42
Track Tutorial Completion 42
Generate Completion Certificate 42
Design Certificate Template 43
Collect Certificate Details from Student 43
Generate Certificate Output 43

Authentication & Access Control **43**

Authenticate Users 43
Seed Initial Admin User 44
Manage the Author Accounts 44
Design Password Reset Flow 44

Analytics Dashboard **44**
Capture Tutorial Events 44
Aggregate Analytics Data 45
Display Analytics Dashboard 45

Collaboration Support **45**
Define Collaboration Model 45
Enforce Edit Locking Functionality 45

Testing & Quality Assurance **46**
Unit Testing 46
Integration Testing 46

Documentation **46**
Author User Guide 46
Student Usage Guide 46
Technical Overview 47

Lack of Knowledge of Technologies Used **50**
Mitigation 50
Monitoring 50
Management 50

Lack of Communication Within the Team **50**
Mitigation 50
Monitoring 50
Management 51

Team Member Does Not Participate Actively **51**
Mitigation 51
Monitoring 51
Management 51

Missing Deadlines **51**
Mitigation 51
Monitoring 51
Management 51

Loss of Work Product **52**
Mitigation 52
Monitoring 52
Management 52

Pressbooks Integration Limitations **52**
Mitigation 52
Monitoring 52
Management 52

Complexity of Branching Tutorial Logic **53**
Mitigation 53
Monitoring 53
Management 53
Scope Expansion from Optional Features **53**
Mitigation 53
Monitoring 53
Management 53
Failure to Deliver a Complete MVP within Sprint Timeframe **53**
Mitigation 54
Monitoring 54
Management 54
Sprint 1 ..... **55**
Summary 55
Tasks 55
Work Products 55
Sprint 2 ..... **56**
Summary 56
Tasks 56
Work Products 56
Sprint 3 ..... **56**
Summary 56
Tasks 56
Work Products 57
Milestone 57
Sprint 4 ..... **57**
Summary 57
Tasks 57
Work Products 57
Sprint 5 ..... **57**
Summary 57
Tasks 58
Work Products 58
Sprint 6 ..... **58**
Summary 58
Tasks 58
Work Products 58
Sprint 7 ..... **58**
Summary 59
Tasks 59
Work Products 59
Final Milestone 59

Vision Statement

The goal of this project is to help librarians, faculty and students at the University of Prince Edward Island (UPEI) Library with a simple, web-based authoring tool for creating high quality learning tutorials. The system will enable its users to easily build, edit and publish interactive and modular tutorials that guide learners through UPEI's library systems, research processes and course-related tasks in a way that is accessible and engaging for all.

The tutorials will be designed to integrate seamlessly with the library's recommended Pressbooks platform, appearing beside course content to provide a contextual "guide on the side". By providing an intuitive and flexible authoring environment, the project aims to increase the effectiveness and reach of skill-building resources, which will in turn help students develop confidence in navigating UPEI's digital library tools and academic materials.

Statement of Scope

The Guide on the Side tool is a web-based application developed for the UPEI Library, designed to present students with a digitally interactive learning experience. The application's primary function is to provide an intuitive authoring interface for librarians and faculty, enabling them to easily create, edit, preview, and publish modular tutorials. These educational resources are built from slides and can incorporate various data objects, including text, images, videos (linked from approved external sources), and fully integrated quizzes to test comprehension. The system's major functionality includes a mechanism for tutorial management, robust support for content embedding, and a critical built-in tool for checking WCAG 2.1 AA accessibility compliance before publication. The application is designed to seamlessly integrate with the Library's recommended Pressbooks platform, allowing the completed tutorials to be easily embedded or linked within course materials. Students will access the final product either directly from the UPEI Library site or via Pressbooks links shared by their instructors. Importantly, the application is constrained by the operational requirement to not collect or store any personal data from students or end-users. The Minimum Viable Product (MVP) will focus on these core authoring features and desktop-first functionality, while features such as analytics or mobile optimization are explicitly out-of-scope for the initial release.

Project Community

The client for this project is the University of Prince Edward Island (UPEI) Library.

The Project Planning Team and Development Team consist of the following members, who are students completing the CS-4810:

- Kadesh Yahaya
- Brian Ezeokeke
- Habeeb Adelagun
- Chinwike Martin Okonkwo

- Luis Mondragon Verduzco
- Bruno Oshiokpekhai

The development team will consist of the students moving on to CS-4820, which means the team will be losing Bruno Oshiokpekhai. Early in the semester the team will be nominating a Project Lead and Technical Lead:

- Kadesh Yahaya
- Brian Ezeokeke
- Habeeb Adelagun
- Chinwike Martin Okonkwo
- Luis Mondragon Verduzco (Comm. Rep.)

Additionally, the development team will also be under the guidance of Dr. David LeBlanc, who will serve in an advisory role with weekly meetings.

Key Requirements:

The key technical requirements for this project are working knowledge of modern web development technologies, specifically **React** for the front-end authoring interface and Node.js with Express for the back-end application logic. The system will be developed as a full-stack web application, requiring team members to be proficient in JavaScript, along with supporting web technologies such as HTML **and** CSS, to implement interactive user interfaces and server-side functionality.

The front-end of the application will be implemented using **React**, which will be used to build the tutorial authoring dashboard, slide editor, and tutorial playback interface. Styling and layout will be handled using standard React-compatible styling approaches to ensure a responsive user experience across modern desktop browsers.

The back end will be implemented using **Node.js and Express**, which will be responsible for managing author authentication, tutorial data storage, publishing workflows, and integration with external platforms. Integration with the **Pressbooks** platform will be achieved through link-based or embedded interactions, allowing tutorials to be displayed alongside Pressbooks content without direct modification of the Pressbooks system.

The system must support the handling and presentation of multiple media types, including formatted text, images, externally hosted videos, and quiz-based interactions, as well as branching logic between tutorial slides. Working knowledge of standard software development tools and practices, including **Git version control** and modern Integrated Development Environments (IDEs) such as **Visual Studio Code**, will be required for effective collaboration and project management.

The key operational requirements for the Guide on the Side project revolve around accessibility compliance, data privacy, and maintenance to ensure the longevity and ethical use of the platform within the UPEI Library system.

The client requires strict adherence to WCAG 2.1 AA accessibility guidelines. This commitment to accessibility affects all operational aspects of the project, including the design of the authoring interface and the final output of the tutorials themselves. Since the core purpose is educational content creation, ease of use and clear documentation for librarians and faculty are of focus to simplify the ongoing process of creating and updating content. The client has indicated that strong documentation, provided in Markdown format, is required for IT administrators to facilitate seamless support and deployment.

Furthermore, a critical constraint involves user privacy: the application must be geared to operate without collecting or storing any personal data from students who consume the tutorials. This ensures the system respects user privacy and avoids complex data compliance issues. Finally, an operational constraint for the Minimum Viable Product (MVP) is the focus on desktop functionality, meaning deployment and testing efforts will be concentrated on ensuring a stable and fully functional experience for users accessing the application via traditional computer environments.

System Acceptance Tests:

The following list comprises the system acceptance tests as discussed with the client (UPEI Library), prioritized based on core functionality and critical constraints:

1. The application successfully allows a Librarian/Faculty user to create, edit, and publish a complete multi-slide tutorial.
2. The published tutorial successfully integrates a working quiz component and embedded external media.
3. The final published tutorial is fully accessible and adheres to the WCAG 2.1 AA guidelines.
4. The system demonstrates successful integration with the Pressbooks platform (e.g., a tutorial is linkable or embeddable in a Pressbooks chapter).
5. Clear documentation in Markdown format is delivered to the client for both librarian content creators and IT administrators.
6. The application demonstrates that no personal data is collected or stored from end-users accessing the tutorials.
7. A complete, functional demonstration tutorial is presented for final testing and review.

Use Cases

1. Create Tutorial

Description:

The Librarian or Faculty member creates and publishes a new interactive tutorial that will be available to students through the Pressbooks system. This involves defining the structure, adding content (text, images, videos, links), and incorporating quizzes.

Actors

- **Librarian / Faculty (Primary):** The author designing the content.
- **System:** The web-based application.
- **Pressbooks:** The external platform where the tutorial is published.

Triggers

- The User selects "Create New Tutorial".

Pre-conditions

- User has authoring access to the **Guide on the Side** tool (via credentials or SSO).

Post-conditions

- Tutorial appears in Pressbooks and is available to users.

Normal Flow

1. The User selects "Create New Tutorial".
2. The System asks the User to input a title, description, and topic.
3. The User adds slides, populating them with text, images, videos, or links.
4. The User adds quiz questions to the appropriate slides.
5. The User reorders or deletes slides as needed.
6. The User previews the tutorial to check the flow and appearance.
7. The User publishes the tutorial to Pressbooks.

Alternate Flows

5A1: User Exits without Publishing

1. After modifying slides (step 5), the User leaves the page or the session times out (**see Use Case 15**).
2. The System stores unsaved progress locally if possible.

3. The tutorial is not published to Pressbooks.

2. Create Tutorial Template

Description:

The Librarian or Faculty member utilizes the simple, web-based interface to build the foundational structure and content for a new interactive tutorial. This process establishes the complete template of slides, quizzes, and media that will be saved, previewed, and eventually published to the live Pressbooks environment.

Actors

- **Librarian / Faculty (Primary):** The author designing and structuring the content.
- **System:** The web-based application interface.

Triggers

- The User clicks the "Create New Tutorial" button on the author dashboard.

Pre-conditions

- The User is successfully logged in (**see Use Case 13**).

Post-conditions

- A new tutorial is saved in the System as an unpublished draft, and the User is ready to add detailed content or publish it.

Normal Flow

1. The User accesses the author dashboard and selects the "Create New Tutorial" option.
2. The System prompts the User to enter basic metadata: a title, description, and primary topic for the tutorial.
3. The User inputs the required metadata.
4. The System loads the blank editing interface, displaying two default slides.
5. The User begins adding the structural elements by:
    - Adding and deleting subsequent slides.
    - Setting the order of the slides.
    - Adding placeholders for text, images, videos, and quizzes to the slides.
6. The User populates the placeholders with content (text, image uploads, video links, and quiz questions).
7. The User selects the "Save Draft" option.
8. The System stores the structured tutorial as an unpublished draft, confirming success to the User.

Alternate Flows

6A1: Accessibility Check Performed

1. After step 6, the User runs the built-in accessibility check.
2. The System highlights any structure or content that violates WCAG 2.1 AA standards.
3. The User adjusts the content/structure to resolve the errors before proceeding to step 7.

7A1: User Exits without Saving

1. After adding content (step 6), the User becomes inactive or closes the browser.
2. The System executes a **Session Timeout** (**see Use Case 15**).
3. If no autosave feature is implemented, the unsaved changes are lost, and the User must start over upon logging back in.

3. Update Tutorial

Description:

The Librarian or Faculty member modifies an existing, previously published tutorial by changing its content, structure, media links, or quizzes, and then pushes the revised version to the live system for students to access.

Actors

- **Librarian / Faculty (Primary):** The author updating the content.
- **System:** The web-based application managing the tutorial drafts and live content.
- **Pressbooks:** The platform displaying the final, updated tutorial.

Triggers

- The User selects an existing tutorial from the "My Tutorials" list to edit.

Pre-conditions

- An existing tutorial has been created and published.
- The User is successfully logged in (**see Use Case 13**) and has edit rights for that tutorial (**see Use Case 12**).

Post-conditions

- The updated tutorial successfully replaces the old version in Pressbooks and is accessible to students.

Normal Flow

1. The User navigates to the "My Tutorials" section on the author dashboard.
2. The User selects the specific tutorial to update.
3. The System loads the tutorial into the editing interface, creating a temporary working draft.
4. The User updates one or more elements of the tutorial, which may include:
    - Modifying text on existing slides.
    - Adding, deleting, or reordering slides.
    - Updating embedded media links (**see Use Case 5**).
    - Adjusting quiz questions or branching logic (**see Use Cases 7, 8**).
5. The User runs an accessibility check to ensure the changes still meet WCAG 2.1 AA standards.
6. The User previews the revised tutorial draft.
7. The User clicks "Republish".

8. The System saves the changes and pushes the new version to replace the old one in Pressbooks.

Alternate Flows

4A1: Content Conflict (See Use Case 4 - Collaborative Editing)
1. While editing (step 4), the System detects that another user has made concurrent changes to the same tutorial.
2. The System alerts the User to the conflict and provides a mechanism to resolve the differences before saving. (**See Use Case 4**)

5A1: Accessibility Check Failure
1. In step 5, the accessibility check fails due to new content or structure changes.
2. The System highlights the errors and prevents the User from proceeding to preview or republish until the accessibility errors are resolved.

7A1: User Exits without Republishing
1. After making changes (step 4), the User becomes inactive or closes the browser before step 7.
2. The System executes a session timeout (**see Use Case 15**) and stores the changes as an un-published draft.
3. The live version of the tutorial remains unchanged in Pressbooks.

4. Collaborative Editing (Future Goal)

Description:

Multiple Librarians and Faculty members can simultaneously or sequentially work on the same tutorial, managing changes and preventing conflicts to ensure seamless teamwork in content creation.

Actors

- **Librarian / Faculty (Primary):** Two or more users with edit rights to the same tutorial.
- **System:** The application's version control and synchronization service.

Triggers

- A second User opens a tutorial that is currently being edited by another User.

Pre-conditions

- The tutorial is published or saved as a draft.
- Both Users have permission to edit the tutorial (**see Use Case 12**).
- The System has been updated to support collaborative features (Future Goal).

Post-conditions

- Changes made by multiple Users are successfully merged or resolved without data loss.

Normal Flow

1. **User A** opens a tutorial for editing.
2. **User B** attempts to open the same tutorial for editing.
3. The System detects that **User A** is already editing the content.
4. The System alerts **User B** that **User A** is currently active and asks if they wish to proceed with concurrent editing or wait.
5. **User B** chooses to proceed with concurrent editing.
6. Both Users make modifications to different parts of the tutorial.
7. **User A** saves their changes.
8. The System automatically synchronizes **User A's** changes.
9. **User B** saves their changes.

10. The System successfully merges **User B's** changes with **User A's** saved version, providing a single, consistent draft.

Alternate Flows

4A1: Lock on Editing

1. In step 4, the System automatically places a soft lock on the tutorial.
2. The System prevents **User B** from editing the tutorial until **User A** saves and closes the session.
3. **User B** is only allowed to view the tutorial in read-only mode.

10A1: Conflict Detection

1. In step 9, both **User A** and **User B** modified the *exact same section* of the same slide.
2. The System detects a conflict during the merge process.
3. The System displays a conflict resolution interface for **User B**, allowing them to accept **User A's** version, keep their own version, or manually merge the text.
4. **User B** resolves the conflict and saves the final version.

5. Embed External Content

Description:

The Librarian or Faculty member integrates outside media, such as videos from approved external sources, into a tutorial slide.

Actors

- **Librarian / Faculty (Primary):** The author adding the media.
- **System:** The tutorial editing interface.

Triggers

- The User selects "Add Embedded Content" while editing a slide.

Pre-conditions

- User is currently editing an existing or new tutorial.

Post-conditions

- Media is successfully viewable inline within the slide or via a direct link.

Normal Flow

1. While editing a tutorial slide, the User selects "Add Embedded Content".
2. The User pastes an external link or media URL (e.g., from YouTube).
3. The System checks if embedding from that source is supported.
4. If embedding is supported, the content shows inline within the slide.
5. If embedding is not supported, a fallback link appears instead.

Alternate Flows

4A1: Embedding Not Supported

1. After step 3, the System determines that embedding the content from the pasted URL is not supported.
2. The System skips step 4 and automatically generates and displays a fallback link for the content (step 5).

6. Embedded File Not Viewable

Description:

The student encounters a slide where the embedded external media (e.g., a video from an approved external source) is missing, blocked, or not loading correctly. The System handles this gracefully by displaying a fallback, allowing the student to continue the learning path.

Actors

- **Student (Primary):** The end-user viewing the tutorial slide.
- **System:** The application displaying the tutorial content and managing media integration.

Triggers

- The Student opens a slide containing embedded external media.

Pre-conditions

- The tutorial contains content embedded from an external source (e.g., YouTube, Vimeo).
- The media file or link has become invalid, has external restrictions, or is blocked by the user's browser/network.

Post-conditions

- The Student can still view the content externally, preventing a blocked learning path.
- The tutorial continues to the next slide.

Normal Flow

1. The Student navigates to a slide that includes embedded external media.
2. The System attempts to load the embedded media file or stream.
3. The System detects an error, restriction, or block preventing the media from loading inline within the slide area.
4. The System hides the broken media player element.
5. The System displays an alternative message (e.g., "Media unavailable. Click the link below.") and/or provides a **direct link** back to the external content's source URL.
6. The Student can choose to click the link to view the media in a new browser tab or simply proceed to the next slide.

Alternate Flows

3A1: Media Loads Slowly

1. In step 3, the System detects a slow load time but no hard error.

2. The System displays a **loading spinner** or similar indicator.
3. If the media eventually loads, the Normal Flow resumes at step 6.
4. If the media times out, the flow continues at step 4.

7. Branching Path

Description:

The Student's progress through the tutorial changes dynamically based on their performance on an integrated quiz, allowing the System to redirect the user to a different slide for remediation (if they answer incorrectly) or an advanced topic (if they answer correctly).

Actors

- **Student (Primary):** The end-user completing the quiz.
- **System:** The application checking quiz answers and managing slide navigation.

Triggers

- The Student submits an answer to a quiz question within a tutorial.

Pre-conditions

- The tutorial author has configured the quiz to include specific branching logic for certain answers.
- The destination slide (remediation or next topic) exists.

Post-conditions

- The Student is successfully navigated to the new slide, and the tutorial flow continues from that point.
- The complete tutorial, including the chosen branch, is explored.

Normal Flow

1. The Student views a slide containing an integrated quiz component.
2. The Student selects and submits an answer to the quiz question.
3. The System checks the answer against the defined branching logic.
4. The System determines the appropriate destination slide based on the result (e.g., correct answer goes to Slide 8; incorrect answer goes to Remediation Slide 5A).
5. The System immediately redirects the Student to the specified destination slide.
6. The Student completes the content in the branched path.
7. The Student continues the rest of the tutorial from the end of the branch.

Alternate Flows

4A1: No Branching Configured

1. In step 3, the System checks the answer but finds **no branching logic** is configured for that specific answer.
2. The System simply provides feedback on the correctness of the answer.
3. The Student must manually click the "Next" button to proceed to the next sequential slide.

5A1: Destination Slide Error

1. In step 5, the destination slide specified in the branching logic has been deleted or is invalid.
2. The System displays an error message to the Student.
3. The System redirects the Student to the **next sequential slide** in the tutorial's main flow, bypassing the intended branch.

8. Create Branching Tutorial Linked to Other Tutorial

Description:

The Librarian or Faculty member creates a tutorial that includes a feature to direct a Student to an alternative slide or section based on their quiz performance (branching path). This is an advanced authoring function that allows for self-remediation or advanced paths. This use case extends the idea to link to other existing tutorials for remediation or additional study.

Actors

- **Librarian / Faculty (Primary):** The author configuring the quiz logic.
- **System:** The web-based application managing quiz answers and links.
- **Student:** The end-user who follows the branching logic.

Triggers

- The User selects an option to configure branching logic after a quiz.

Pre-conditions

- The tutorial contains a quiz component.
- Other relevant tutorial(s) exist and are published.

Post-conditions

- The quiz question successfully redirects the Student to the intended internal slide or an external link to another tutorial upon completion.

Normal Flow

1. The User is in the editing view for a tutorial (**see Use Case 3**).
2. The User navigates to a slide containing a quiz question.
3. The User selects the option to configure branching logic for the quiz answers.
4. For a specific answer (e.g., an incorrect one), the User configures the redirection destination.
5. The User can choose to redirect to: a. An existing slide within the current tutorial (internal branch, per **Use Case 7**). b. A link to a different, published tutorial (external link branch).
6. The User saves the branching logic and previews the tutorial to test the redirection.

7. The User republishes the updated tutorial (**see Use Case 3**).

Alternate Flows

5bA1: External Link is Invalid

1. In step 5b, the User inputs a link to an external tutorial that is broken or no longer published.
2. The System alerts the User during the preview (step 6) that the link is invalid.
3. The User must update or remove the link before republishing.

9. User views and completes tutorial

Description:

The User accesses and completes a published tutorial, moving through slides and interacting with embedded content and quizzes.

Actors

- **User (Primary):** The end-user consuming the content.
- **System:** The tutorial display interface.

Triggers

- The User opens the tutorial link or finds it in Pressbooks.

Pre-conditions

- The tutorial has been created and published by the author.

Post-conditions

- The Tutorial is finished; no personal data is saved.

Normal Flow

1. The User opens the tutorial link or finds it in Pressbooks.
2. The slides appear in the correct order.
3. The User moves through slides using Next and Previous controls.
4. The User interacts with quizzes and media components.
5. The User completes all sections.
6. The User downloads a completion certificate.

Alternate Flows

4A1: Branching Path Activated (Use Case 7)

1. While interacting with a quiz (step 4), the User answers a question.
2. The System checks the answer.
3. The System redirects the User to a slide that matches the result.
4. The User completes the branch and returns to the main path.

4A2: Embedded Content Not Viewable (Use Case 6)

1. While viewing a slide with media (step 4), the System detects an error or restriction with the embedded content.
2. The System shows an alternative message or a fallback link.

3. The User can still access the content externally if desired.

10. Generate Certificate (Stretch Goal)

Description:

After successfully completing all required sections of a tutorial, the Student or Librarian generates a PDF certificate to document the achievement. This is considered a stretch goal for the project's Minimum Viable Product (MVP).

Actors

- **Student / Librarian (Primary):** The user who completes the tutorial and requests the certificate.
- **System:** The application managing tutorial completion status and PDF generation.

Triggers

- The User finishes all slides and/or quiz requirements for a tutorial.

Pre-conditions

- The System has verified the tutorial's completion status.
- This feature has been developed and enabled.

Post-conditions

- A PDF certificate is downloaded and saved locally to the user's device.

Normal Flow

1. The User completes all required content and quizzes within a tutorial (**see Use Case 9**).
2. The System recognizes and verifies that the tutorial's completion criteria have been met.
3. The System displays a notification or a **"Generate Certificate"** button.
4. The User clicks the "Generate Certificate" button.
5. The System creates a PDF document, dynamically including the **UPEI branding**, the **tutorial name**, and the **completion date**.
6. The System prompts the User to download the certificate.
7. The PDF is downloaded and saved to the User's local device.

Alternate Flows

2A1: Completion Criteria Not Met

1. In step 2, the System determines that one or more quiz sections were not fully completed.

2. The System informs the User that the certificate cannot be generated and highlights the incomplete sections.
3. The User is prompted to review the missing sections.

11. Analytics Dashboard (Optional/Future Goal)

Description:

The Librarian or Faculty member accesses a dashboard to view aggregate statistics and key performance indicators related to student engagement with and performance on the published tutorials, allowing them to measure effectiveness and identify areas for improvement.

Actors

- **Librarian / Faculty (Primary):** The user reviewing the data.
- **System:** The application's web interface and data tracking component.

Triggers

- The User selects the "Analytics" option from the author dashboard.

Pre-conditions

- The System has been updated beyond the MVP to include the analytics component.
- The System has successfully tracked and stored student usage data (views, quiz attempts, completion).
- The User has appropriate access permissions to view the analytics dashboard.

Post-conditions

- The User has successfully accessed and reviewed usage statistics for their tutorials.

Normal Flow

1. The User successfully logs into the author dashboard (**see Use Case 13**).
2. The User selects the "Analytics Dashboard" link.
3. The System retrieves and displays key high-level metrics for all published tutorials (e.g., total views across all content, overall completion rate).
4. The User selects a specific tutorial from a list to view detailed reports.
5. The System displays in-depth statistics for the selected tutorial, such as:
    - Completion rates.
    - Average time spent per slide.
    - Success/failure rates for specific quiz questions.
6. The User reviews the data to identify necessary content updates or structural improvements.

Alternate Flows

3A1: No Data Available

1. In step 3, the System attempts to retrieve data but finds no tracked usage history.
2. The System displays a message indicating that data is not yet available, prompting the User to wait for student activity.

5A1: Detailed Data Filtered

1. After step 5, the User selects a filter option (e.g., date range, specific course).
2. The System re-queries the data and refreshes the display to show only the metrics matching the specified filter.

12. Manage Access and Permissions (Optional)

Description:

IT Staff or a designated Librarian manages access to the authoring dashboard by assigning specific roles (e.g., **author, editor, viewer**) to users (Librarians and Faculty), thereby restricting the features available to them to maintain security and control over content creation.

Actors

- **Librarian / IT Staff (Primary):** The user responsible for account administration and assigning roles.
- **System:** The application managing user roles and permissions.

Triggers

- IT Staff initiates the account review process or a new user requests authoring access.

Pre-conditions

- The **Guide on the Side** tool has been deployed and user accounts are established (e.g., through UPEI's network authentication).
- This feature is enabled (as it is Optional).

Post-conditions

- Only approved users can edit or publish tutorials based on their assigned roles.
- The user database is updated with the correct permission levels.

Normal Flow

1. The Librarian or IT Staff member logs into the administrative zone of the System.
2. The User accesses the interface for managing user accounts and permissions.
3. The User reviews the existing list of faculty and staff accounts.
4. The User selects a specific account and assigns a role (e.g., **author, editor, or viewer**).
5. The System stores these permission levels in the user database.
6. When a User logs in (**see Use Case 13**), the System references these stored permissions to limit or enable available features (e.g., preventing a 'viewer' from seeing the "Republish" button).

Alternate Flows

4A1: Role Downgrade Conflict

1. In step 4, the User attempts to downgrade the role of an **Author** who is currently editing a tutorial (**see Use Case 4**).
2. The System presents a warning that the user is currently active and prevents the role change until the active session ends (**see Use Case 15**).

5A1: Failed Permission Retrieval

1. The System successfully stores the permissions (step 5) but fails to retrieve them during a User's login attempt (Normal Flow step 6).
2. The System defaults to the most restrictive permission level (e.g., read-only access) and logs the event for administrative review.
3. The System displays an alert to the User about the temporary access restriction.

13. Log In

Description:

The Librarian or Faculty member securely accesses the web-based author dashboard to create and manage tutorials.

Actors

- **Librarian / Faculty (Primary):** The user attempting to access the authoring tools.
- **System:** The application's authentication service.

Triggers

- The User navigates to the authoring tool URL.

Pre-conditions

- The User has valid credentials (username/password) or UPEI Single Sign-On (SSO) is enabled and available.
- This feature is enabled (as it is Optional).

Post-conditions

- User is successfully logged in, and the author dashboard is loaded, ready for work.

Normal Flow

1. The User opens the web page for the **Guide on the Side** authoring tool.
2. The System presents the login page.
3. The User enters their login credentials.
4. The System authenticates the User (potentially via UPEI SSO if enabled).
5. If authentication is successful, the System loads the author dashboard.
6. The User is now able to access creation and management features.

Alternate Flows

4A1: Authentication Failure

1. In step 4, the System fails to authenticate the User (e.g., incorrect password).
2. The System displays an "Authentication Failed" message.
3. The User is prompted to re-enter their credentials or use the "Forgot Password" function (**see Use Case 16**).

4A2: User Not Authorized (See Use Case 14)

1. In step 4, authentication is successful, but the System determines the User lacks the necessary permissions (e.g., is not a designated author).
2. The System displays an "Access Denied" page instead of the dashboard.
3. The User is prompted to contact an administrator for role assignment.

14. Not Authorized (Optional)

Description:

The System prevents an **Unauthenticated User** (a non-logged-in person) or a **User with Insufficient Permissions** (e.g., a student trying to access the author dashboard) from gaining access to restricted pages or features.

Actors

- **Unauthenticated User / User with Insufficient Permissions (Primary):** The person attempting unauthorized access.
- **System:** The application's security and access control mechanism.

Triggers

- The User attempts to navigate directly to a restricted URL (e.g., the authoring dashboard, an administrator settings page).

Pre-conditions

- The user is either not logged in, or is logged in but lacks the appropriate role (e.g., is not a Librarian or Faculty author).
- The System has security measures in place to define restricted pages.
- This feature is enabled (as it is Optional).

Post-conditions

- Unauthorized access is prevented.
- The User is redirected to a safe, permissible page (e.g., the login screen or a public tutorial).

Normal Flow

1. The User attempts to access a page requiring a specific permission level (e.g., the tutorial editor).
2. The System checks the User's current session or credentials.
3. The System determines that the User's current access level is **Insufficient** for the requested page.
4. The System immediately denies access.

5. The System displays an "Access Denied" page or redirects the User to the **Login screen** (**see Use Case 13**).

Alternate Flows

5A1: Logged-In User Redirected

1. After step 4, the User is currently logged in as a **Student** but tries to access the author dashboard.
2. The System determines the user is unauthorized for that specific page.
3. The System redirects the User to the publicly available UPEI Library site or the **Tutorial Viewing** interface (**see Use Case 9**).

15. Session Timeout

Description:

The System automatically logs out an inactive Librarian or Faculty member after a predefined period of inactivity. This is a critical security measure designed to protect the authoring dashboard and prevent unauthorized changes.

Actors

- **Librarian / Faculty (Primary):** The user whose session is being monitored.
- **System:** The application's session management service.

Triggers

- The User remains inactive (no mouse movement, key presses, or server requests) for the set duration of the session timeout.

Pre-conditions

- The User is currently logged into the authoring dashboard (**see Use Case 13**).
- A session timeout duration is configured in the System.

Post-conditions

- The user session is terminated safely.
- The User is logged out and returned to the login screen.

Normal Flow

1. The User successfully logs in and begins work on the authoring dashboard.
2. The User becomes inactive (stops interacting with the interface).
3. The System continuously monitors the time elapsed since the last user interaction.
4. Once the inactivity period exceeds the configured timeout duration, the System triggers the logout process.
5. The System attempts to store any unsaved progress locally if possible.
6. The System automatically logs the User out.
7. The System redirects the User to the login screen, requiring re-authentication to resume work.

Alternate Flows

5A1: Unsaved Changes Detected

1. In step 5, the System detects unsaved changes to a tutorial or settings.
2. The System presents a small, final pop-up warning the User that the session is about to expire and asking if they want to discard or attempt a quick save (if technically feasible).
3. If no action is taken, the session times out, and the draft remains unsaved on the server.

4A1: Pre-Timeout Warning

1. Just before the official timeout (e.g., 60 seconds prior), the System displays a warning pop-up to the User (e.g., "Your session will expire in 60 seconds due to inactivity. Click OK to stay logged in.").
2. If the User interacts with the warning, the timer is reset, and the Normal Flow restarts at step 2.
3. If the User ignores the warning, the Normal Flow continues at step 5.

16. Forgot Password (Optional)

Description:

The Librarian or Faculty member recovers their lost account password by initiating a reset process via the email address associated with their account.

Actors

- **Librarian / Faculty (Primary):** The user attempting to recover a lost password.
- **System:** The application's account recovery service (which may interface with the UPEI network).

Triggers

- The User clicks the "Forgot Password" link on the login page.

Pre-conditions

- The User has a registered and working email address associated with an author account in the System.
- This feature is enabled (as it is Optional).

Post-conditions

- The User successfully creates a new password and access to the author dashboard is restored.

Normal Flow

1. The User navigates to the login page and clicks the **"Forgot Password"** link.
2. The System asks the User to enter the registered email address associated with their account.
3. The User submits the email address.
4. The System verifies the email address against its user database.
5. If verified, the System sends a **password reset link** to that email address.
6. The User receives the email and clicks the reset link.
7. The System loads a secure page prompting the User to create a new password.
8. The User creates and confirms a new password, which the System saves.
9. The User then logs in successfully using the new credentials.

Alternate Flows

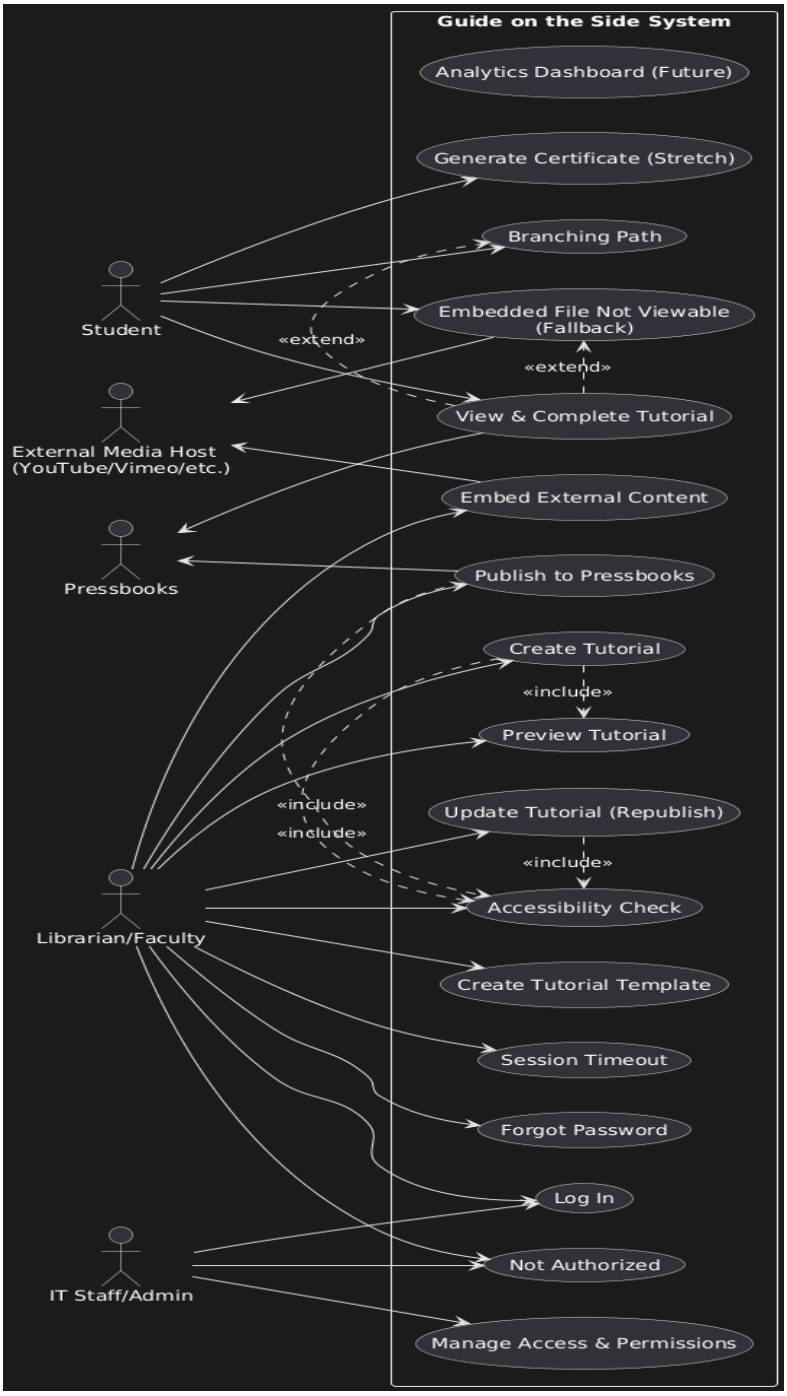
4A1: Email Address Not Found

1. In step 4, the System fails to find the submitted email address in the user database.
2. The System displays an error message stating the email is unrecognized, without confirming whether the account exists (for security reasons).
3. The User is prompted to re-enter the email address.

5A1: Reset Link Expires

1. In step 5, the reset link is sent but has a time limit (e.g., 2 hours).
2. The User attempts to use the link after the time limit has passed.
3. The System displays an error, invalidates the link, and prompts the User to restart the "Forgot Password" process.

Use Case Diagram



The use case diagram provides a high-level overview of the Guide on the Side system. It identifies the primary actors and illustrates their interactions with the system's core

functionalities. The diagram serves as a visual summary of the use cases described in the previous section and establishes the foundation for the activity and sequence diagrams that follow.

Feature List

Research & Setup

Identify Pressbooks Integration Mechanisms

Description:
Determine how the guide-on-the-side tool can be integrated into Pressbooks, including available extension points, embedding options, and platform limitations.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
Provision of a short document describing supported integration approaches and constraints within Pressbooks.

Determine Two-Pane Tutorial Presentation Model

Description:
Identify how tutorial instructions and embedded Pressbooks content will be displayed side-by-side, including layout behavior and navigation constraints.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
Provision of a documented explanation of how the two-pane layout will be presented to students.

Identify Required Authoring Capabilities

Description:
To ensure that high quality tutorials are made, it is essential to identify all the author actions required to create and manage tutorials, including slide creation, text editing, question configuration, and branching.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
A finalized list of authoring actions that the system must support.

Evaluate Rich-Text Editing Options

Description:
Evaluate the available options for allowing authors to enter formatted instructional text within slides.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
One rich-text editing approach is selected and documented with justification as to why we chose it.

Define Authentication and Role Requirements

Description:
Define user roles (admin, author) and clarify the authentication, authorization, and password recovery requirements.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
A documented role and authentication model agreed upon by the team.

Identify Analytics Requirements for Dashboard

Description:
Determine what tutorial usage data must be collected to feed the analytics dashboard for authors.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
We must have a list of required analytics events and metrics.

Establish a Development Workflow

Description:
Define the git repository structure, the branching strategy, testing expectations, and local setup process, to ensure that all team members can properly contribute.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 3 hours

Acceptance Criteria:
All team members can successfully clone the repository and run the project locally using the documented steps.

Tutorial Data Representation

Define Tutorial Representation

Description:
Define what data constitutes a tutorial, including metadata, status (like draft, published, unpublished or archived), and ownership.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
A well-documented tutorial data structure is generated.

Define Slide Representation

Description:
Define how slides are structured, ordered, and associated with the created tutorials.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
A well-documented slide representation model is generated.

Define Question Representation

Description:
Define question types, answer options, and feedback mechanisms within slides.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
A well-documented question representation model must be generated.

Define Branching Representation

Description:
We must define how branching rules link answers to slides or tutorials, for proper flow-control.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
A well-documented branching representation model must be created.

Tutorial Creation & Management Services

Create the Tutorial Service

Description:
Functionality to create a new tutorial in draft state with assigned ownership must be provided.

Priority: 1 (very high)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
A new draft tutorial can be created and stored successfully by authors.

Retrieve the Tutorial Service

Description:
Provide functionality to retrieve tutorial data, including slides and questions, with permission checks.

Priority: 1 (very high)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
All authorized users should successfully be able to load full tutorial data.

Update the Tutorial Service

Description:
Provide functionality to update tutorial metadata and content.

Priority: 1 (very high)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
Updates made to tutorials must persist correctly and reflect upon reload.

List the Tutorials Service

Description:
Provide functionality to list tutorials owned by an author with basic filtering capabilities.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
Tutorials are listed correctly when selected filters are applied.

Authoring Interface

Display the Author Dashboard

Description:
Display a dashboard listing an author's tutorials and available actions.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
Authors must be able to view their created tutorials, select them and perform chosen actions on them.

Create a Tutorial UI Flow

Description:
Provide UI for entering tutorial details and beginning the tutorial authoring process.

Priority: 1 (very high)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
Submitting the form creates a draft tutorial and opens the editor to allow the authors make desired changes to a tutorial.

Allow Authors to Manage Slides

Description:
It is essential to allow the authors to add, remove, and reorder slides so they properly fit into the desired context.

Priority: 1 (very high)
Effort Required: 3 (medium)
Estimated Time: 8 hours

Acceptance Criteria:
The slide order chosen by the author updates correctly and persists.

Allow Authors Edit Slide Content

Description:
Allow the authors to edit slide titles and formatted instructional text.

Priority: 1 (very high)
Effort Required: 3 (medium)
Estimated Time: 8 hours
Acceptance Criteria:
The slide content must save and reload properly.

Allow the Authors to Manage Questions

Description:
Allow authors to add and edit questions and answer options to slides.

Priority: 2 (high)
Effort Required: 3 (medium)
Estimated Time: 8 hours
Acceptance Criteria:
Questions display correctly in preview and playback mode.

Allow authors Configure Branching Rules

Description:
Allow authors to configure navigation paths based on question answers, to ensure they all fit into context.

Priority: 2 (high)
Effort Required: 3 (medium)
Estimated Time: 8 hours
Acceptance Criteria:
Branching behavior functions as configured.

Allow Authors Preview Tutorial

Description:
Allow authors to preview tutorials in student view without publishing, to have a good idea of the final tutorial before publishing.

Priority: 3 (medium)
Effort Required: 2 (low)

Estimated Time: 4 hours
Acceptance Criteria:
Preview matches student experience.

Undo / Redo Support

Description:
Allow authors to undo and redo recent edits during slide editing.

Priority: 3 (medium)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
The undo/redo system should restore recent states reliably.

Autosave Editing State

Description:
Provide a system that automatically save current editing state at intervals.

Priority: 3 (medium)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
System restores the latest autosaved state upon refresh.

Tutorial Playback (Student Experience)

Render Two-Pane Layout

Description:
Provide a system to render instructional content alongside embedded Pressbooks content.

Priority: 1 (very high)
Effort Required: 3 (medium)
Estimated Time: 8 hours
Acceptance Criteria:
The system layout displays slides and embedded content correctly for tutorials.

Allow students Navigate Slides

Description:
Provide a system to allow students to navigate slides in order.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
The navigation works reliably without skipping any required steps or slides.

Display Questions & Record Response

Description:
Provide a system to display the tutorial questions and collect student responses.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 6 hours
Acceptance Criteria:
The student responses should be correctly captured.

Execute Branching Logic

Description:
Provide a system to navigate the tutorial flow based on student answers.

Priority: 2 (high)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
The branching paths execute as configured by the author.

Track Tutorial Completion

Description:
Determine when a tutorial is completed and show a completion screen.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
The tutorial completion state is correctly detected and verified.

Certificate Generation

Generate Completion Certificate

Description: Provide the ability to generate a completion certificate after a student finishes a tutorial (if enabled for that tutorial).

Priority: 3 (medium)
Effort Required: 3 (medium)
Estimated Time: 5 hours
Acceptance Criteria: After completion of a tutorial, the student can generate a certificate that includes tutorial title and completion date.

Design Certificate Template

Description: Define the certificate layout and required fields (tutorial name, date, student name).
Priority: 3 (medium)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria: A documented certificate template design is approved by the team/client.

Collect Certificate Details from Student

Description: Provide a simple UI prompt to capture optional student information (e.g., name) before generating the certificate.
Priority: 3 (medium)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria: A student can enter details and proceed to the certificate generation process.

Generate Certificate Output

Description: Generate the certificate in a printable/exportable format.
Priority: 3 (medium)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria: Certificate can be downloaded or printed successfully and matches the template.

Authentication & Access Control

Authenticate Users

Description:
Provide a login and logout functionality.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
The system must allow authenticated users to access protected features.

Seed Initial Admin User

Description:
Provide an initial administrator account for system management.

Priority: 1 (very high)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
Admin user can log in successfully.

Manage the Author Accounts

Description:
Provide a system to allow the admins to create and manage author accounts.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
New authors can be created and assigned roles by the admin.

Design Password Reset Flow

Description:
Implement a system to allow users to reset forgotten passwords.

Priority: 2 (high)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
Users can reset passwords successfully via a secure token flow.

Analytics Dashboard

Capture Tutorial Events

Description:
Implement a system to correctly capture tutorial views, progression, and completions.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
The underlined tutorial events are correctly captured and stored.

Aggregate Analytics Data

Description:
Implement a system to summarize the captured events into metrics on the dashboard.

Priority: 3 (medium)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
Aggregated metrics are accurate.

Display Analytics Dashboard

Description:
Design the dashboard to correctly display charts and tables using real usage data.

Priority: 3 (medium)
Effort Required: 3 (medium)
Estimated Time: 8 hours
Acceptance Criteria:
The system should display at least one chart and one table display using real data.

Collaboration Support

Define Collaboration Model

Description:
Design a system to define how concurrent edits are handled.

Priority: 3 (medium)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
A chosen collaboration model by the team is documented.

Enforce Edit Locking Functionality

Description:
Implement a system to prevent conflicting simultaneous edits.

Priority: 3 (medium)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
Only one author can make an edit to a tutorial at a time.

Testing & Quality Assurance

Unit Testing

Description:
Upon creation of tutorial and authoring functionality, they must be tested to ensure that they work as expected.

Priority: 2 (high)
Effort Required: 3 (medium)
Estimated Time: 8 hours
Acceptance Criteria:
All the tests pass for the core services.

Integration Testing

Description:
Design a system to test end-to-end author and student flows.

Priority: 2 (high)
Effort Required: 3 (medium)
Estimated Time: 6 hours
Acceptance Criteria:
The complete author and user flows function as expected.

Documentation

Author User Guide

Description:
Create documentation for tutorial authors, to enable them learn how to make use of the system.

Priority: 2 (high)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
Upon making use of the documentation, authors have a good understanding of the system and can follow the guide to create tutorials.

Student Usage Guide

Description:
Create documentation for the students using tutorials.

Priority: 3 (medium)
Effort Required: 2 (low)
Estimated Time: 3 hours
Acceptance Criteria:
Upon making use of the documentation provided, students can navigate tutorials without assistance.

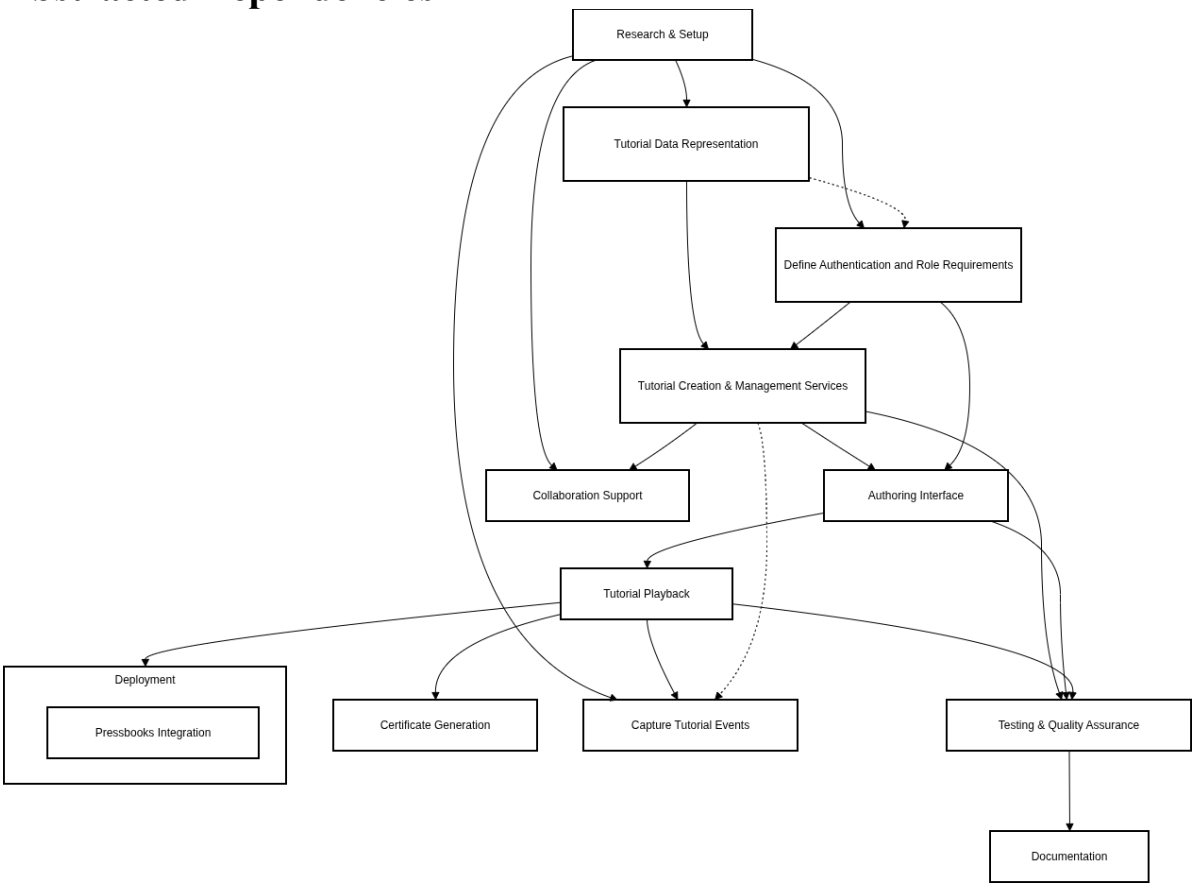
Technical Overview

Description:
Document the system architecture and integration points.

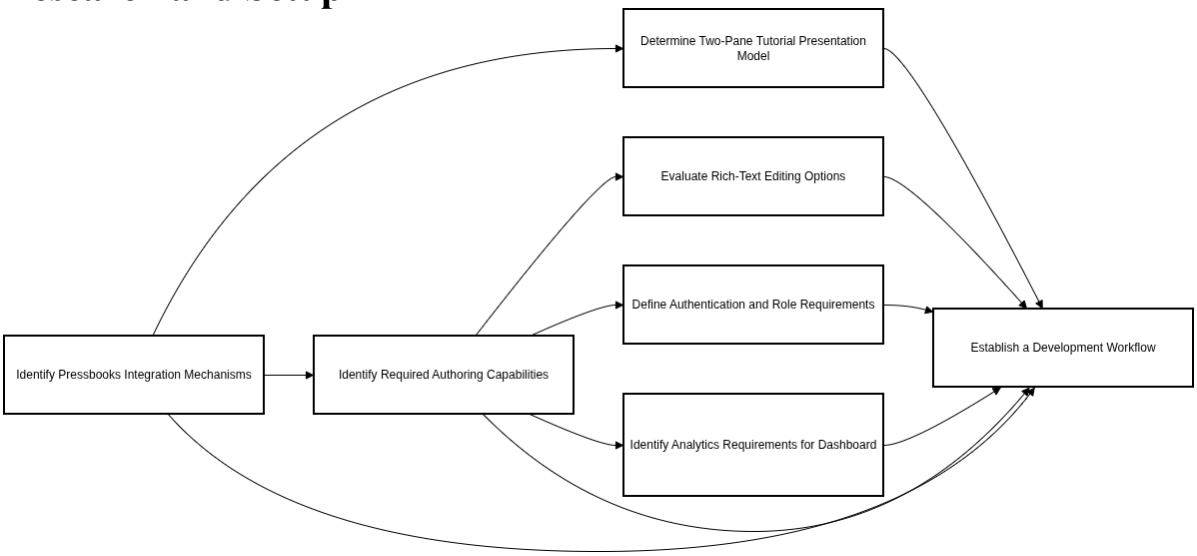
Priority: 3 (medium)
Effort Required: 2 (low)
Estimated Time: 4 hours
Acceptance Criteria:
The documented technical overview accurately reflects system.

Dependency Charts

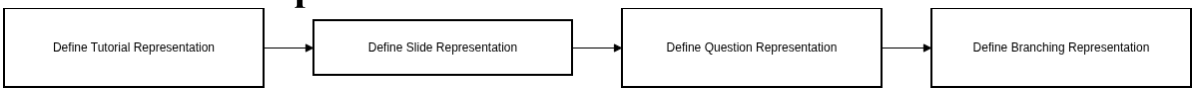
Abstracted Dependencies



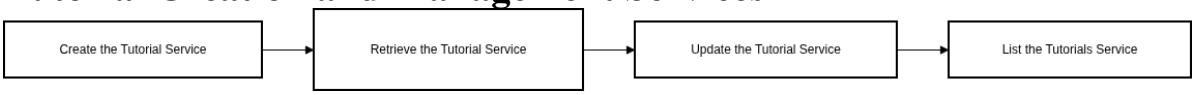
Research and Setup



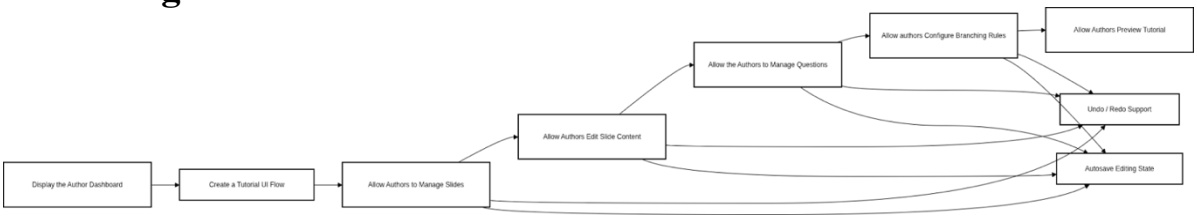
Tutorial Data Representation



Tutorial Creation and Management Services

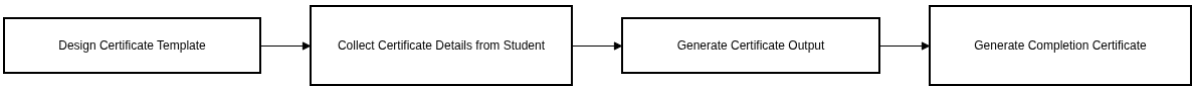


Authoring Interface

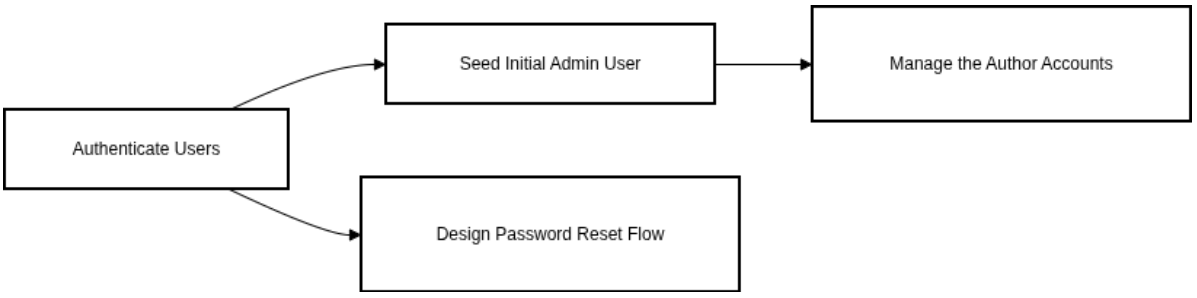


Tutorial Playback

Certificate Generation



Authentication and Access Control



Analytics Dashboard

RMMM Plan

Lack of Knowledge of Technologies Used

The group members may not have sufficient prior experience with the technologies used in the project, such as the outlined web frameworks, authentication mechanisms, or third-party integrations.

Mitigation

- Team members will research unfamiliar technologies using online resources and documentation.
- Tasks will be assigned based on individual strengths and familiarity with specific technologies.
- Frequent standups can help to get weaker team members get up to speed in understanding and implementation.

Monitoring

- Team members will assess their progress regularly.
- Difficulties experienced will be communicated during team meetings.

Management

- The team lead will provide guidance and reassign tasks if necessary.
- Additional learning resources will be shared within the team whenever applicable.

Lack of Communication Within the Team

Poor communication among the team members may lead to misunderstandings, duplicated work, or incomplete features.

Mitigation

- Regular team meetings will be scheduled.
- A group chat will be created to constantly check in on group members.
- Communication channels will be clearly defined.
- Responsibilities will be documented frequently.

Monitoring

- Attendance and participation in meetings will be collected.
- Task progress will be reviewed regularly.

Management

- The team lead will address communication gaps among group members.
- Responsibilities will be clarified when confusion arises.

Team Member Does Not Participate Actively

A team member or team members may not contribute sufficiently due to scheduling conflicts or lack of engagement.

Mitigation

- Responsibilities will be clearly assigned.
- Expectations will be communicated early.
- Frequent feedback to Dr Leblanc will motivate team members

Monitoring

- Contribution to tasks and meetings will be tracked.
- Deliverables will be reviewed per group member.

Management

- Tasks may be reassigned if necessary.
- The issue will be escalated to the instructor if unresolved in time.

Missing Deadlines

Tasks may not be completed within expected timeframes due to underestimation, unexpected issues or just running behind schedule.

Mitigation

- The tasks will be broken into smaller units.
- Reasonable deadlines will be set.

Monitoring

- Progress will be reviewed during meetings.
- Early warnings will be raised whenever a delays occurs.

Management

- Deadlines may be adjusted if justified.

- Team members may assist each other to recover schedule slippage.

Loss of Work Product

Project builds may be lost due to system failure, accidental deletion, or versioning mistakes from group members.

Mitigation

- Version control systems will be used consistently.
- Frequent commits will be encouraged.
- Clear and concise commit messages will be enforced

Monitoring

- Repository activity will be monitored.
- Team members will confirm backups exist.

Management

- Recovery will be attempted using version history.
- Processes will be reinforced to prevent recurrence.

Pressbooks Integration Limitations

There might be unexpected limitations in embedding or integrating with Pressbooks may restrict functionality or require design changes.

Mitigation

- Integration approaches will be researched early.
- Design will remain flexible to accommodate any limitations.

Monitoring

- Integration experiments will be reviewed regularly (smoke testing the code but probably less frequently).
- Issues will be documented as they arise.

Management

- Simplified integration approaches will be adopted if required.
- Non-essential features will be deferred or removed at the client’s permission if necessary.

Complexity of Branching Tutorial Logic

Branching paths based on quiz responses may introduce logical errors or inconsistent tutorial flows.

Mitigation

- Branching rules will be clearly defined.
- Edge cases will be considered thoroughly during design.

Monitoring

- Tutorial flows will be reviewed during testing.
- Peer testing will be performed.

Management

- Branching behavior will be simplified if needed.
- Incorrect paths will be corrected promptly.

Scope Expansion from Optional Features

Optional features such as analytics, certificates, or collaboration may consume more time than expected.

Mitigation

- Core features will be prioritized.
- Optional features will be clearly labeled and assigned as such.

Monitoring

- Feature progress will be tracked per sprint.
- Time spent on optional features will be reviewed.

Management

- Optional features may be deferred or reduced.
- Focus will be redirected to completing the MVP.

Failure to Deliver a Complete MVP within Sprint Timeframe

Due to accumulated risks, the project may not reach a functional minimum viable product.

Mitigation

- MVP scope will be clearly defined.
- Development will focus on core functionality first, before optional features.

Monitoring

- The proximity to MVP readiness will be reviewed after each phase.
- Critical feature completion will be tracked.

Management

- MVP functionality will be stabilized before adding enhancements.
- Remaining work will be prioritized to ensure delivery.

Delivery Plan

Based on the priority of the tasks, the dependencies identified in the feature list, and the risks outlined in the RMMM plan, especially risks related to unfamiliar technologies, Pressbooks integration constraints, and scope expansion, the following delivery schedule has been developed.

The schedule is based off the assumption that the project will be completed by 5 team members, each contributing approximately 18 hours per week. Using two-week sprints, this provides approximately 180 development hours per sprint. Due to expected estimation inaccuracies and learning overhead associated with the selected web technologies, a buffer of approximately 50% has been applied to each sprint. As a result, each sprint is planned to include 90–100 hours of scheduled work, leaving flexibility to absorb delays.

Lower-priority and optional features are intentionally scheduled in later sprints so that they may be deferred if necessary to ensure timely delivery of a functional Minimum Viable Product (MVP).

Sprint 1

Jan 12 – Jan 25

Summary

Research, environment setup, Pressbooks integration investigation, and foundational design decisions.

Tasks

1. Set up the project repositories on git and establish branching strategy
2. Configure the development environments for frontend and backend
3. Research Pressbooks embedding and integration constraints
4. Define two-pane tutorial layout behavior
5. Identify required authoring capabilities
6. Evaluate rich-text editing libraries for tutorial content
7. Define authentication and user role requirements
8. Define tutorial, slide, question, and branching representations
9. Establish development workflow and coding standards
10. Draft an initial technical overview documentation

Work Products

- Documentation describing Pressbooks integration approach
- Defined tutorial, slide, and question data models
- Selected a rich-text editing solution

- Repository with working development up and running
- Initial technical documentation

Sprint 2

Jan 26 – Feb 8

Summary

Core tutorial creation services and basic authoring interface.

Tasks

1. Implement tutorial creation service (draft tutorials)
2. Implement tutorial retrieval service
3. Implement tutorial updation service
4. Implement tutorial listing service
5. Display the author's dashboard
6. Implement the "Create Tutorial" UI flow
7. Implement slide list display and reordering
8. Implement basic slide content editing
9. Implement draft save behavior
10. Initial unit tests for tutorial services

Work Products

- Functional tutorial CRUD services
- Author dashboard with a tutorial list
- Basic tutorial editor with slide management
- Test cases for core tutorial services

Sprint 3

Feb 9 – Feb 22

Summary

Question authoring, branching configuration, and tutorial preview.

Tasks

1. Implement question creation and editing
2. Implement answer options and validation
3. Implement branching rule configuration
4. Implement branching data persistence
5. Implement tutorial preview mode

6. Validate the tutorial flow in preview
7. Refine slide editing interface
8. Create additional unit tests for questions and branching

Work Products

- Question and branching authoring functionality
- Preview mode matching student experience
- Updated test coverage for branching logic

Milestone

The authoring MVP complete, authors can create, edit, preview, and configure branching tutorials. At this stage, the product shoud be presented to the client to get some feedback

Sprint 4

Feb 23 – Mar 8

Summary

Student tutorial playback and completion tracking.

Tasks

1. Implement two-pane tutorial playback layout
2. Implement slide navigation for students
3. Display questions and collect responses
4. Execute branching logic during playback
5. Implement tutorial completion detection
6. Implement fallback handling for embedded content
7. Integration testing of playback flow

Work Products

- Fully functional student tutorial playback
- Branching behavior executed correctly
- Completion state reliably detected

Sprint 5

Mar 9 – Mar 22

Summary

Authentication, access control, and analytics foundations.

Tasks

1. Implement user login and logout
2. Seed initial admin user
3. Implement author account management
4. Implement password reset flow
5. Capture tutorial usage events
6. Aggregate analytics data
7. Implement basic analytics dashboard
8. Documentation for author workflows

Work Products

- Authentication and role-based access control
- Functional analytics dashboard (basic metrics)
- Create a user guide (Markdown / .md files)

Sprint 6

Mar 23 – Apr 5

Summary

Optional features, collaboration controls, and quality improvements.

Tasks

1. Define collaboration model
2. Implement tutorial edit locking
3. Implement undo/redo support
4. Implement autosave functionality
5. Implement certificate generation (stretch goal)
6. Accessibility review and improvements
7. Expanded integration testing

Work Products

- Edit locking and autosave mechanisms
- Undo/redo functionality
- Optional certificate generation
- Accessibility compliance refinements

Sprint 7

Apr 6 – Apr 19

Summary

Stabilization, documentation, and final testing.

Tasks

1. End-to-end system testing
2. Bug fixes and refinements
3. Final accessibility validation
4. Technical overview documentation
5. Finalize student and author guides
6. Prepare final project demonstration

Work Products

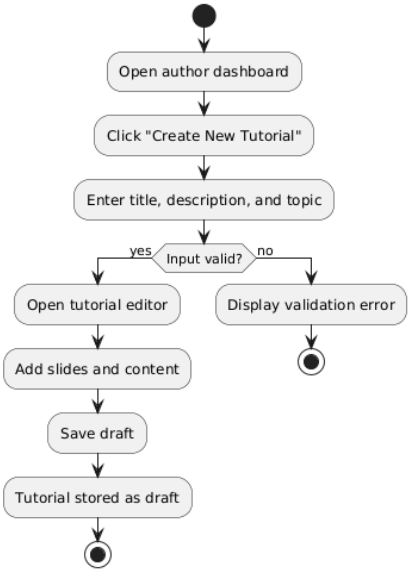
- Fully tested system
- Complete documentation set
- Final presentation/demo materials

Final Milestone

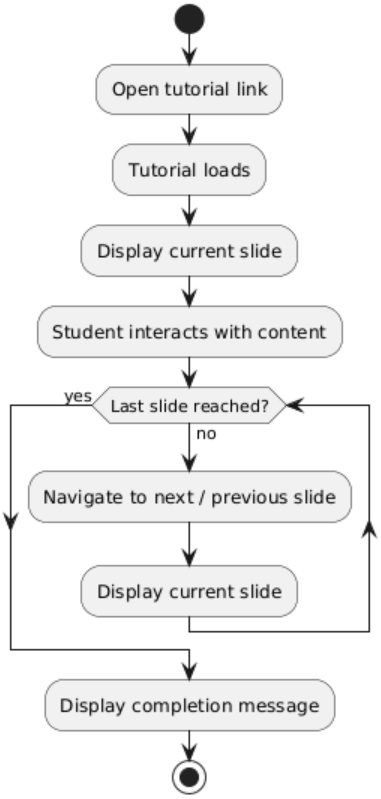
A complete Guide on the Side system is delivered to the client being the UPEI Library, supporting tutorial authoring, student playback, branching logic, analytics, and documentation, ready for evaluation and demonstration.

Activity Diagrams

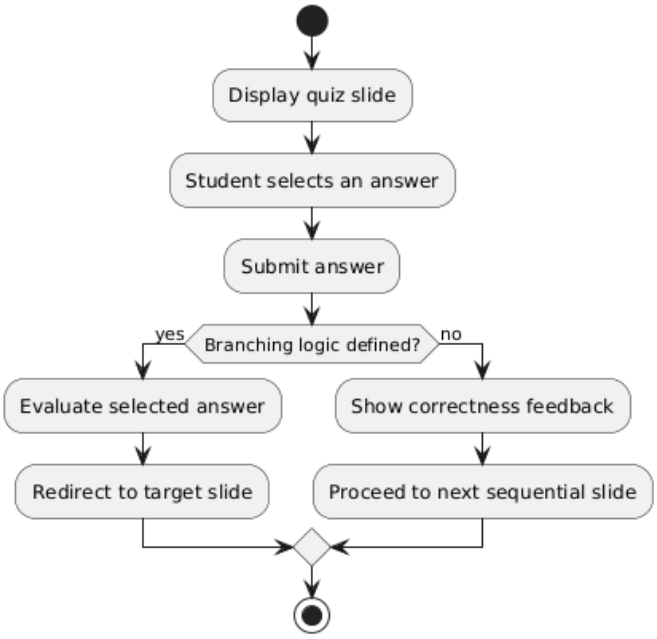
Create new Tutorial



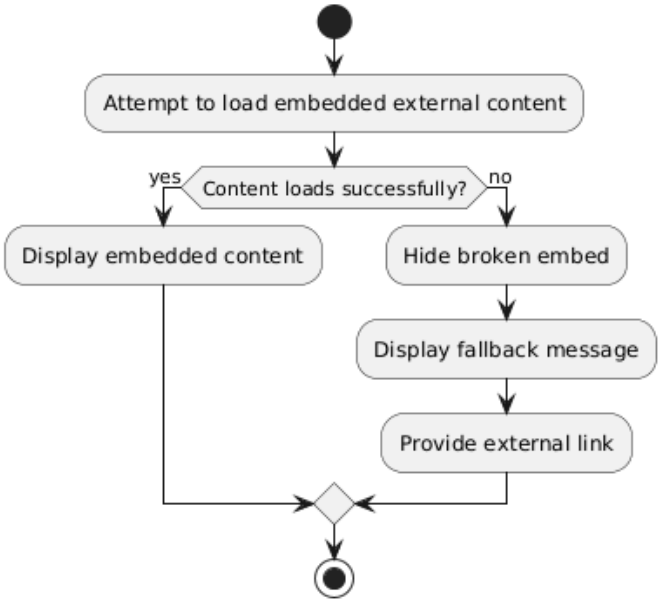
View & Complete Tutorial



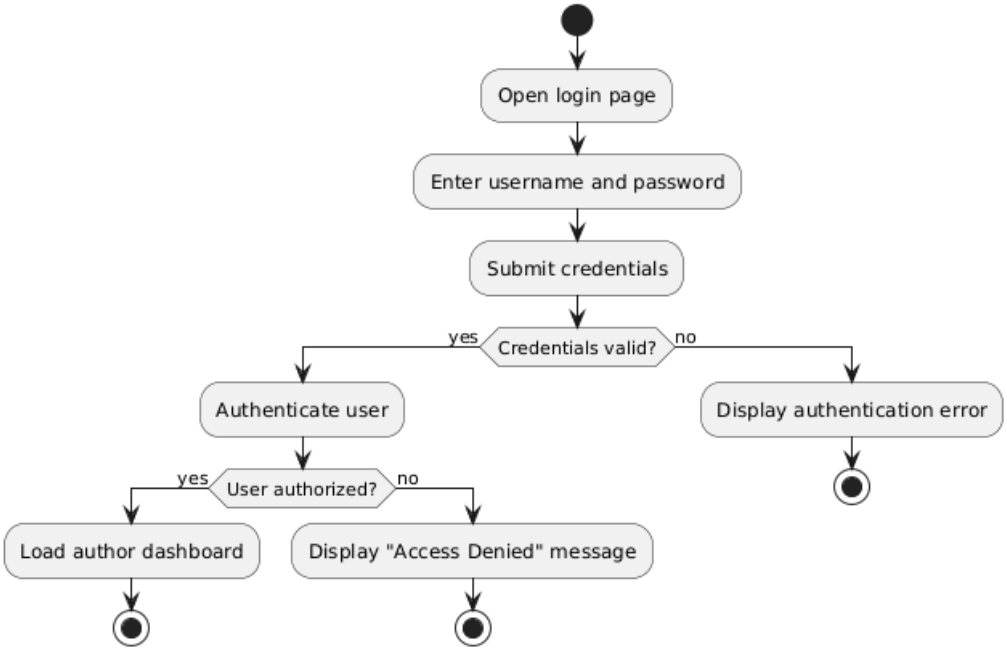
Branching Path



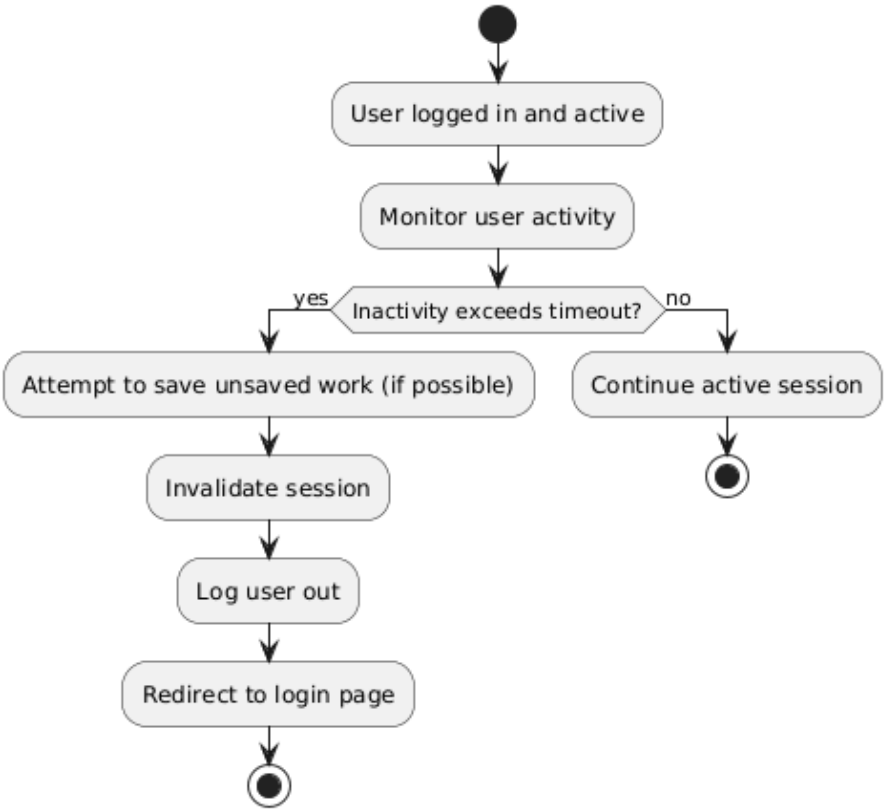
Embedded Path Not Viewable (Fallback)



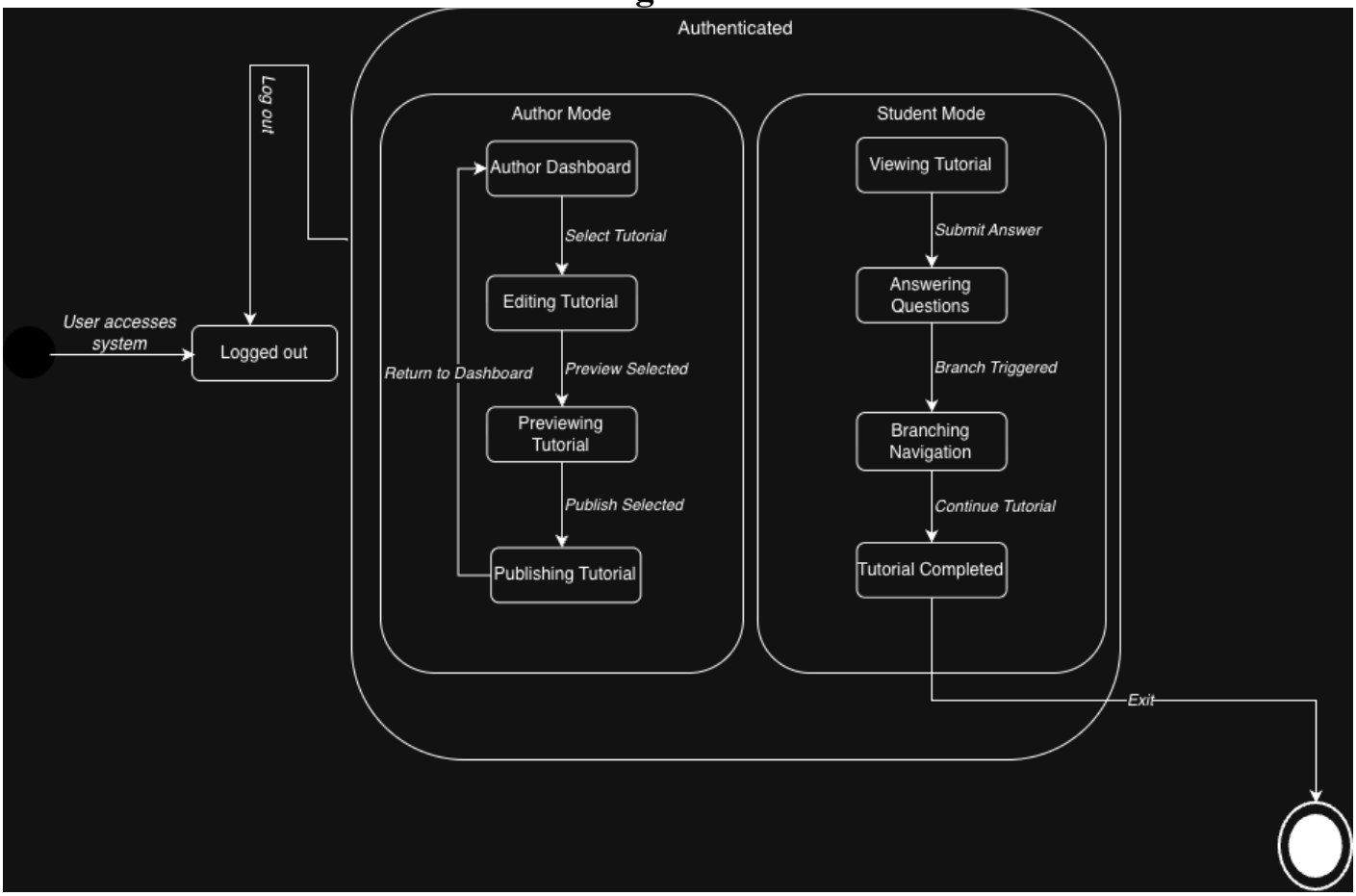
Login Authorization



Set Timeout

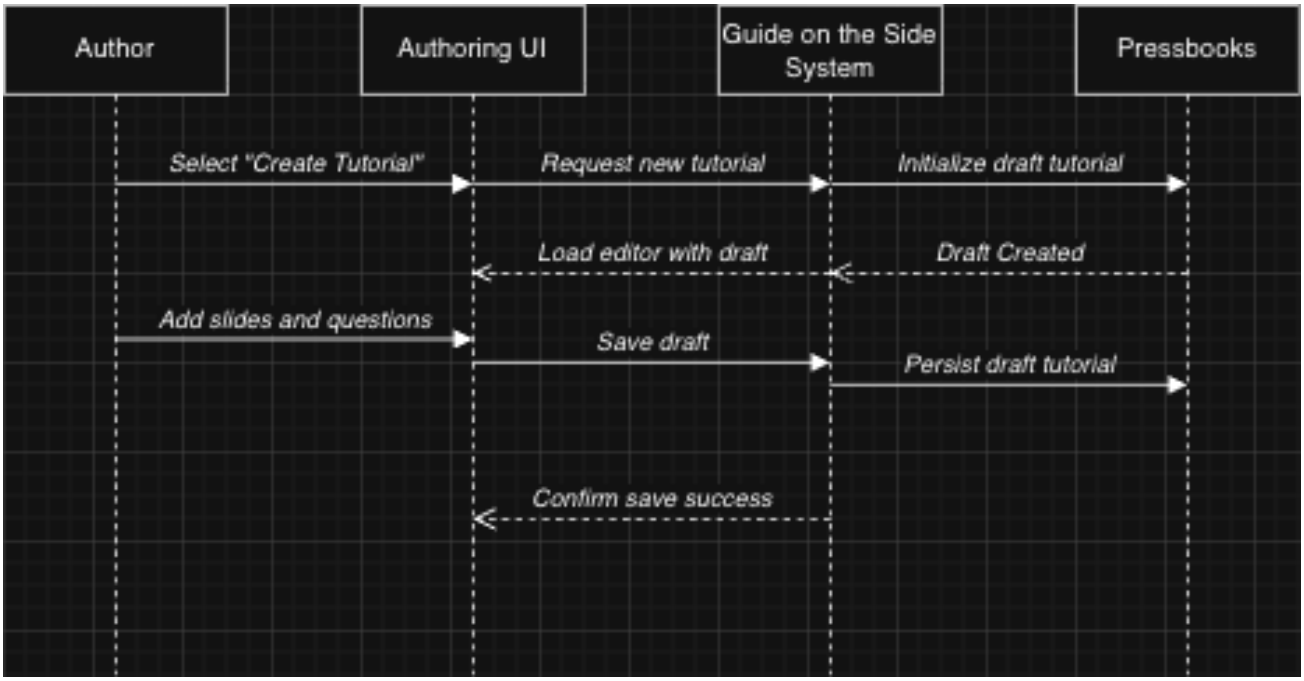


State Diagram

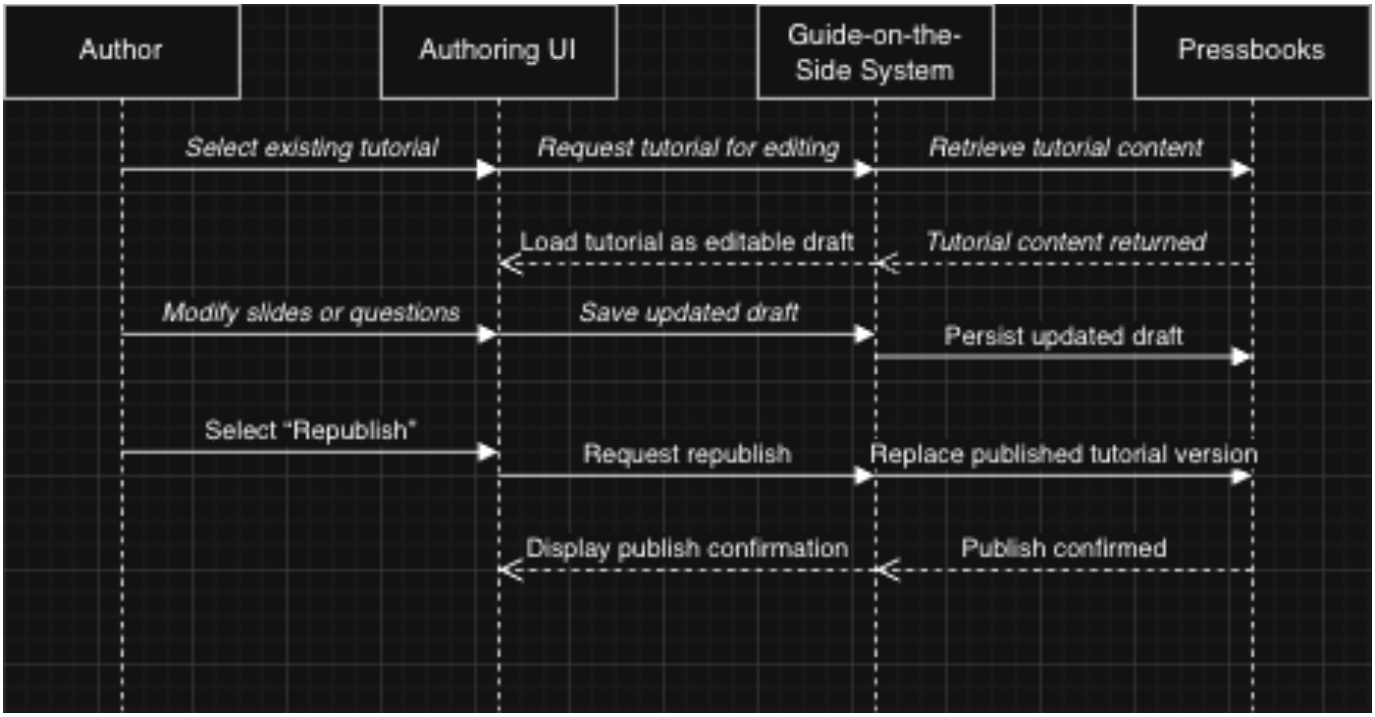


Sequence Diagrams

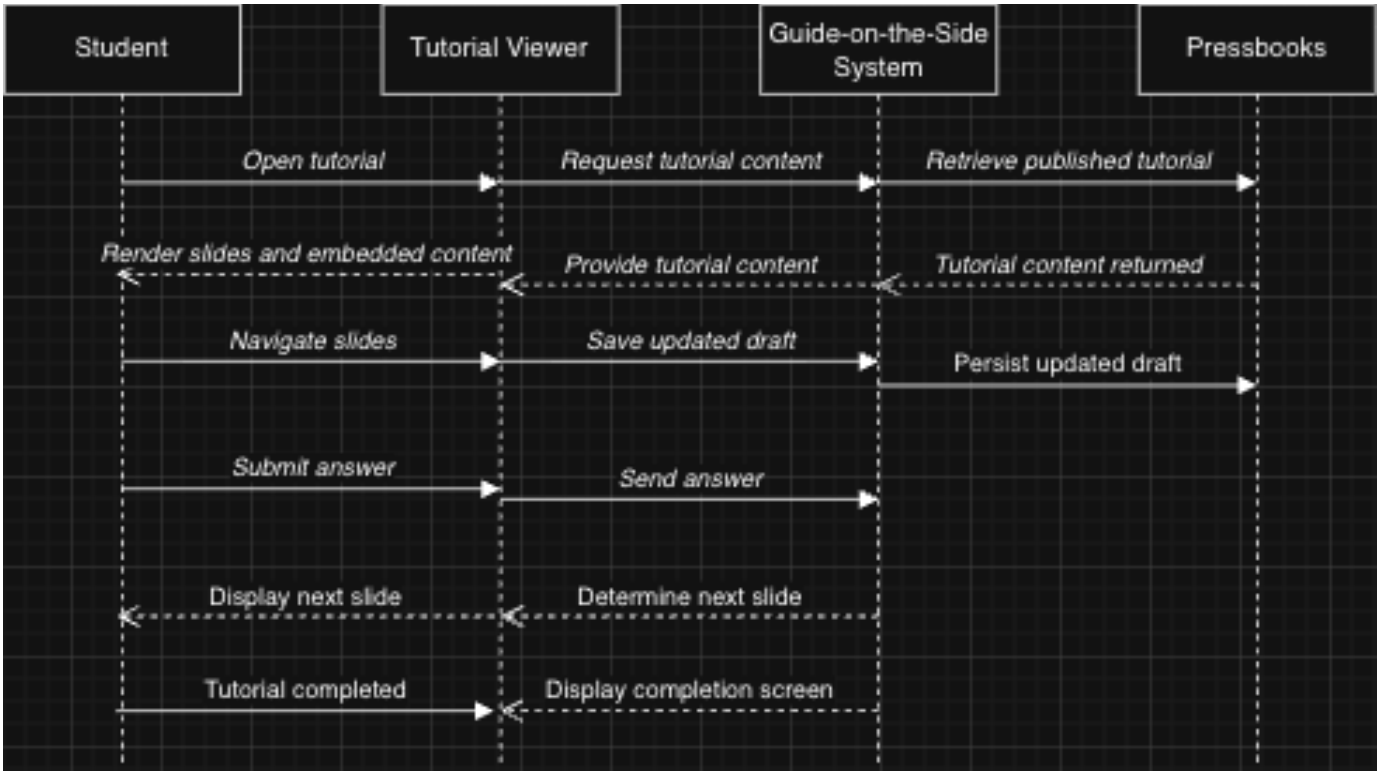
Create Tutorial



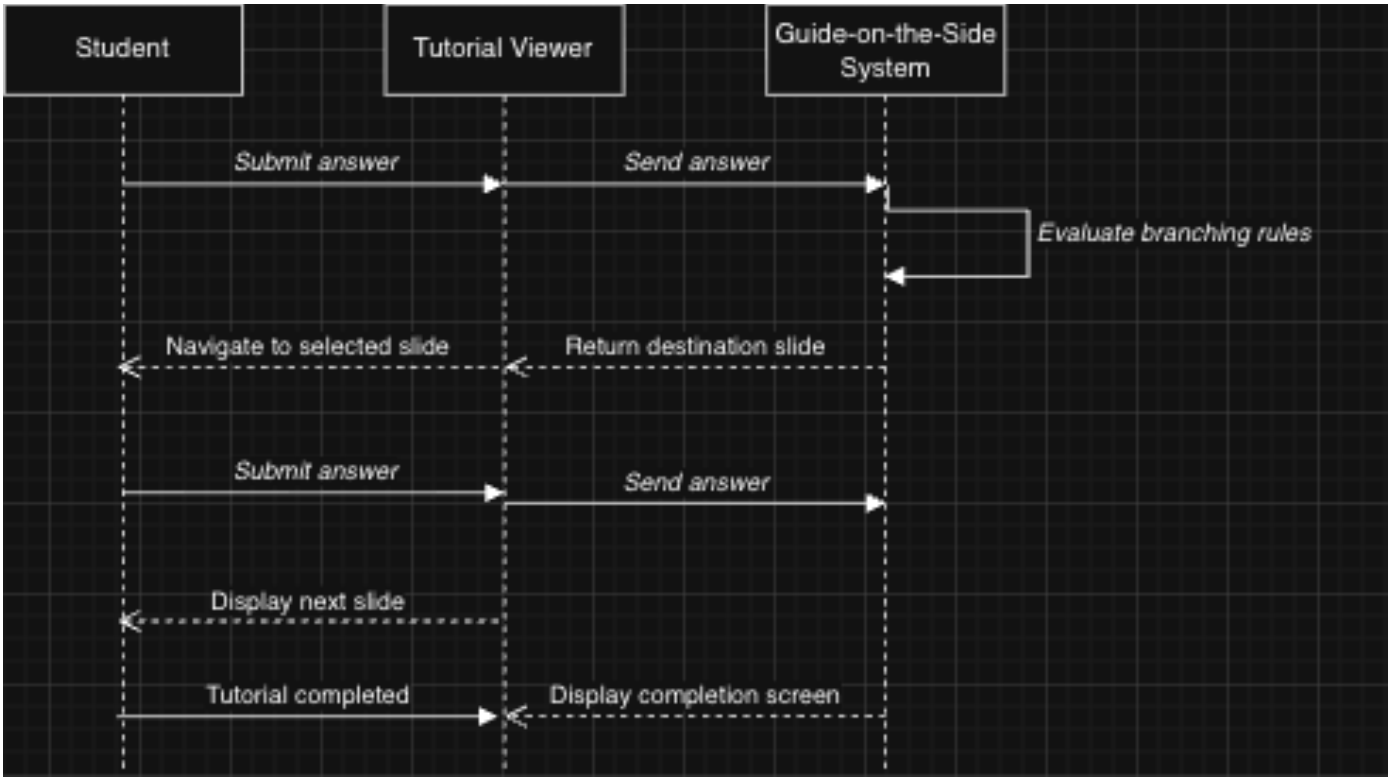
Update Tutorial



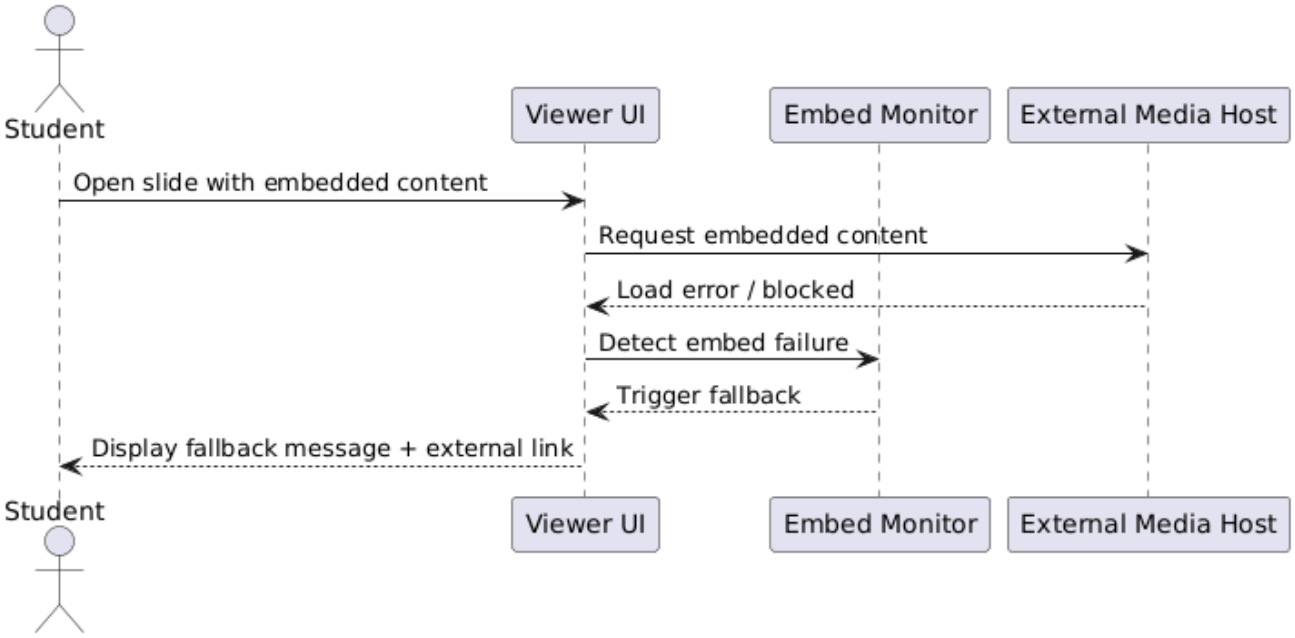
View and Complete



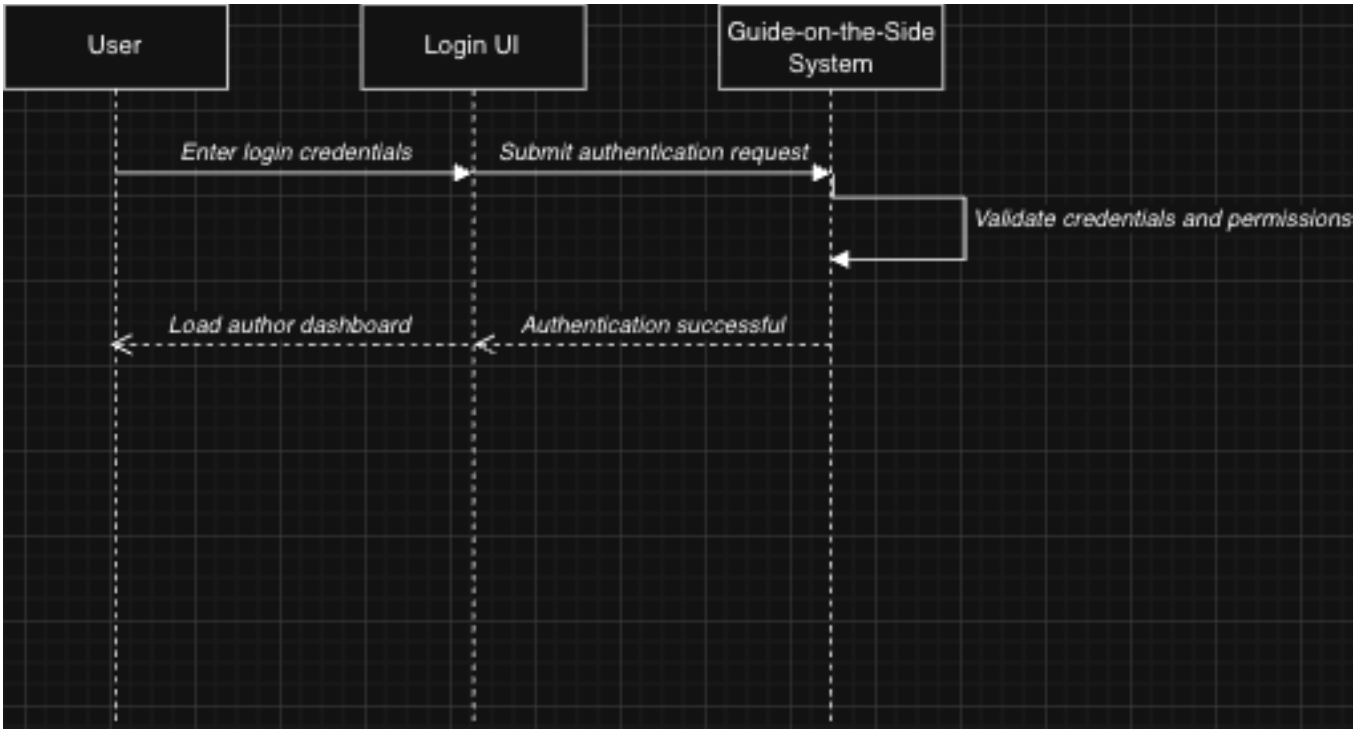
Branching Path



Embedded Path Not Available



Login Not Authorized



Session Timeout