CS Department Automated Information Timeline
Iteration 1 Submission

Matthew Hays, Pawan Bhandari, Sarah Faron, Tim Klimpel

February 5, 2024

Contents

List of Figures

1 Vision Statement

2 Requirements

2.1 Functional Requirements

REQ01: Faculty members can create a post for review, which will initially default to a proposed state until approved.
REQ02: An admin/reviewer must approve a post before it can be published to the site.
REQ03: A user can add events to an event calendar that can be displayed on screen.
REQ04: A user can add pictures to posts.
REQ05: A user can add videos to posts.
REQ06: A user can add HTML pages to posts.
REQ07: Created posts that will be displayed on a large format screen.
REQ08: The system will allow management of which posts are displayed.
REQ09: The system will allow management of which media is displayed.
REQ10: The system will fall back to the 10 most recent posts if no posts are tagged, or less than 10 posts are tagged.
REQ11: The Office manager will use the WYSIWYG editor to modify static content posts.
REQ12: The Office Manager will use the WYSIWYG editor to tag posts for display on screen.
REQ13: The system will allow audiences to use their mobile devices (e.g., through a QR code that brings them to the website) to browse all content beyond the 10 most recent posts.
REQ14: The system will allow audiences to use their mobile device to browse the calendar.
REQ15: A user will authenticate and be assigned a valid role or view the content as a non-authenticated user.

2.2 Non-Functional Requirements

REQ16: The system will have the following user roles: Faculty, Admin, and Office Manager.
REQ17: Content rolling and updates to content will be completed via WebSockets.
REQ18: The backend will consist of a Spring Framework REST API with a database and appropriate middleware.
REQ19: The system will manage a media library which consists of all posted pictures and videos.
REQ20: The system will consist of a front-end display: a television screen.
REQ21: The system will consist of a front-end display: a web-based display optimized for all screen sizes.
REQ22: The design of the user interface must utilize Baylor's official color scheme.

- Baylor Green: #154734
- University Gold: #FFB81C
- Secondary and accent colors can be chosen from the lists provided here.

REQ23: The system will have a WYSIWYG editor.

3 Glossary

The following terms are important to the development of the project:

- **WYSIWYG Editor:** What You See Is What You Get editor
- **WebSocket:**
- **Post:** A picture, video, or manually created html page.
- **Page:**
- **Event Calendar:** A calendar which is displayed on screen which faculty can add events.
- **Media Library:** A library which stores both picture and video content.
- **Faculty:** Admin, Reviewer, Officer Manager, and other university employee.
- **Admin:** A privileged user that holds all access rights.
- **Reviewer:** A user that acts as a reviewer of posts to approve or reject posts that have been submitted for review.
- **Officer Manager:** A faculty member that holds special privilege over the system under discussion to manage which posts are displayed at any given time.

4 Domain Model

The below Domain model represents a Draft version of the current planned domain model. Classes/objects are noted with a $\_D$ to indicate they are *draft* classes.

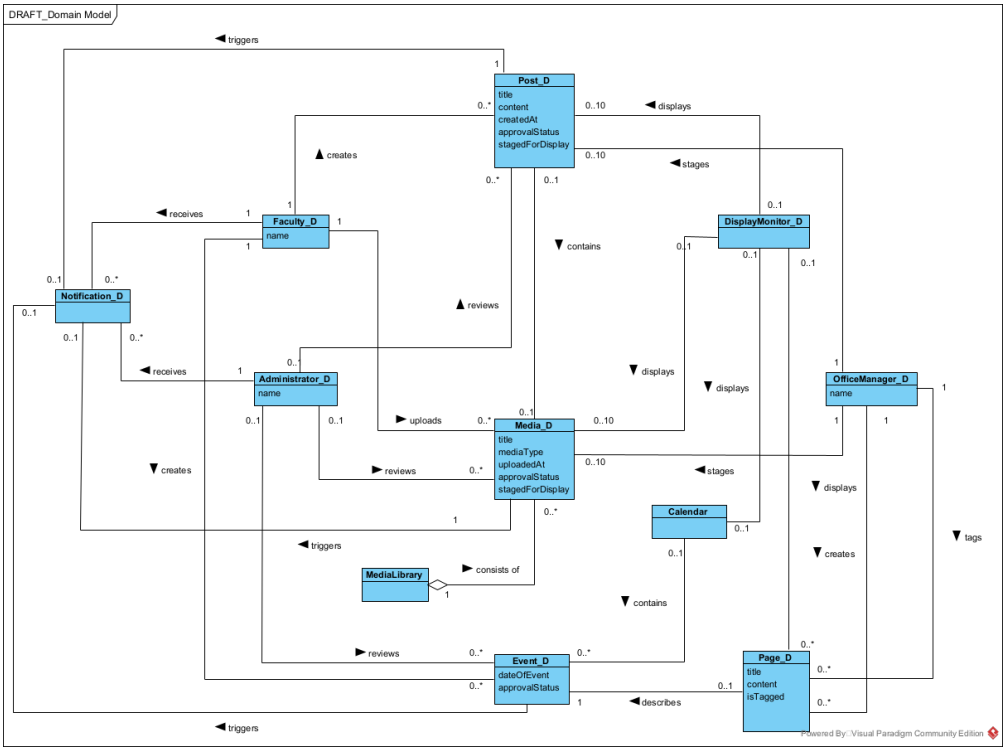

Figure 1: Domain Model for CS Department Automated Information Timeline

5 Use Cases

5.1 Use Case Diagram

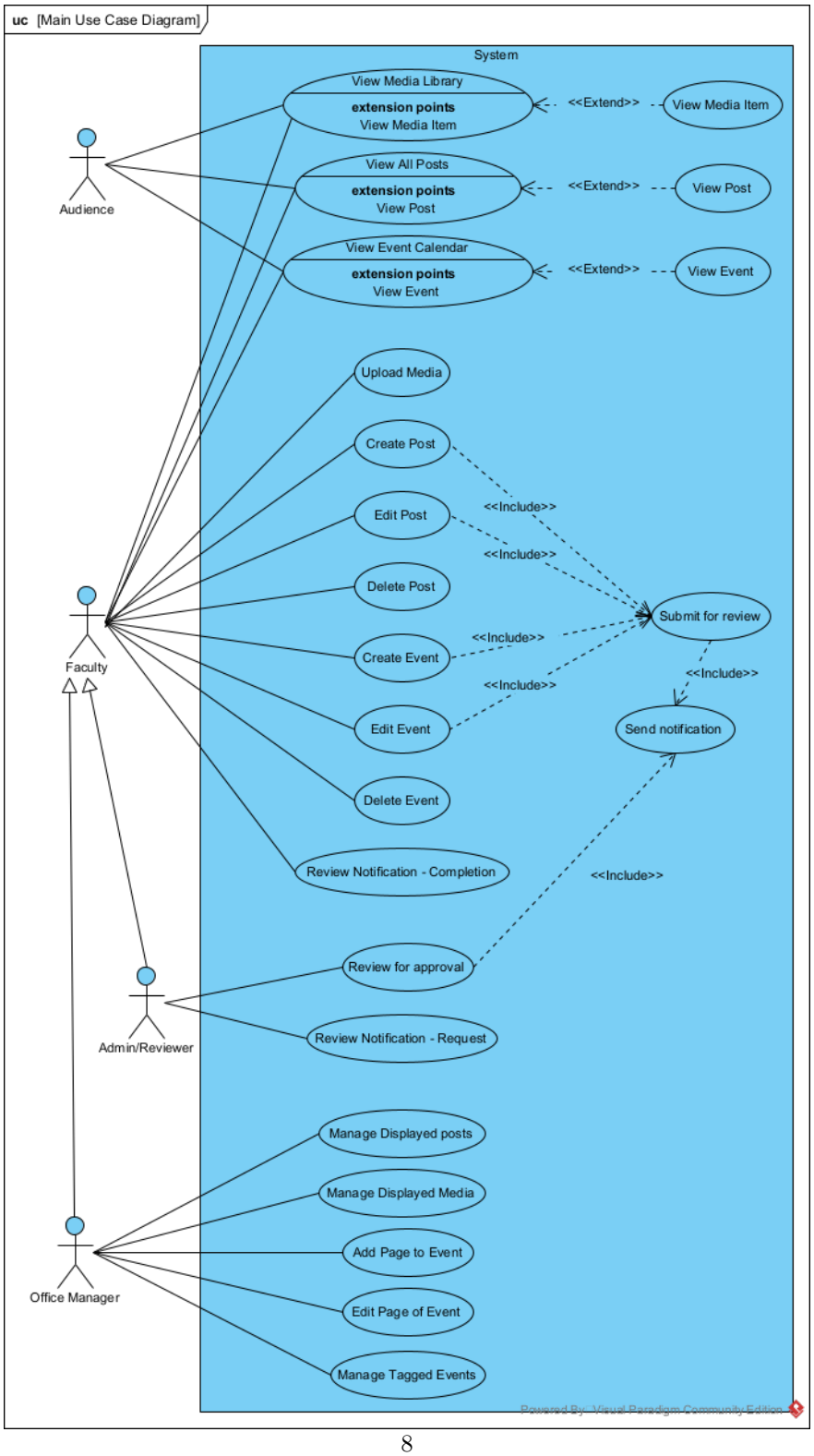

Figure 2: Use Case Diagram for CS Department Automated Information Timeline

5.2 Use Cases

There are a total of 16 use cases written at this stage in the project. Each team member was assigned 3 use cases to fully analyze. Each of these 12 use cases are written in fully-dressed form and accompanied by System Sequence Diagrams (SSDs) and Operations Contracts (OCs). The remaining 4 use cases are written in casual/brief form.

The assigned use cases are as follows:

Bhandari	Faron	Hays	Klimpel
UC11	UC05	UC04	UC15
UC12	UC06	UC09	UC16
UC13	UC07	UC10	UC01

5.2.1 UC01: View Media Library

ID: UC01 (View Media Library)
Scope: CS Automated Information Timeline
Level: User goal
Primary Actor: Audience
Stakeholders and Interests:

- Audience wants the ability to view all shared media through the media library
- Faculty wants to be able to allow guests to view media shared by them
- Office Manager wants to ensure that all media is available to be seen through the web portal
- Admin wants to be sure that approved media can be displayed to the audience

Preconditions:

- Media library contains approved media items

Postconditions:

- Media library is shared with the audience

Main Success Scenario:

1. Audience member scans available QR code
2. System provides appropriate URL for main home page
3. Audience member navigates to URL
4. Audience member selects “Media library” option
5. System presents full media library view

Alternative Flows:
3A: URL is down

1. Audience member alerts faculty or office manager
2. Office manager or faculty review system status

5A: Multiple pages available

1. System presents paged view with limit of items
2. Audience member selects 'next page'
3. System returns next set of results
4. Continue until end of media library is returned

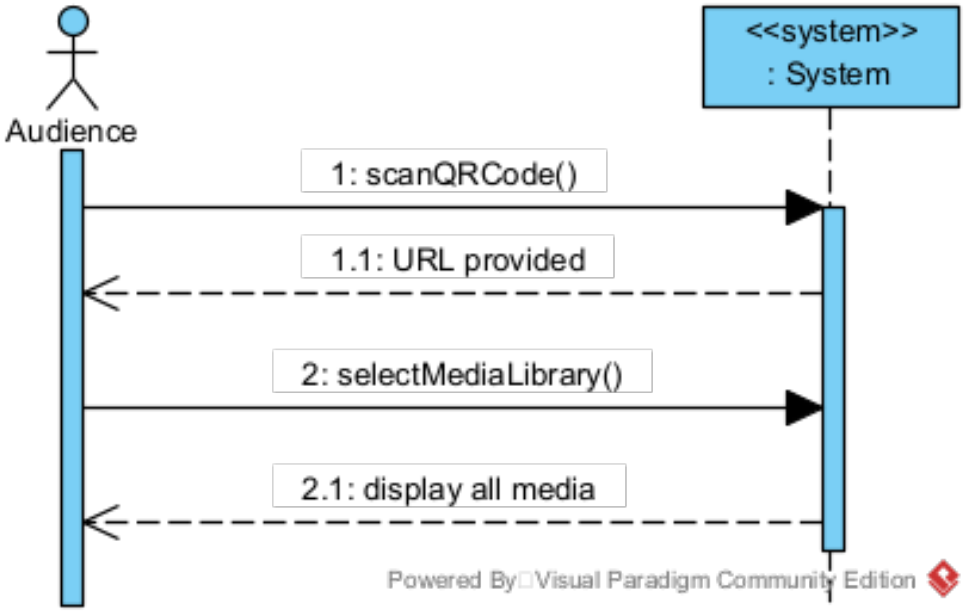

Figure 3: System Sequence Diagram: View Media Library

Operation: selectMediaLibrary()
Cross-References: UC01 (View Media Library)
Pre-conditions:

- Media items, *m*, exists in system
- *m* is approved for view

Post-conditions:

- List of media, *ml*, is created
- *ml* is returned to user

5.2.2 UC02: View All Posts

Brief.

5.2.3 UC03: View Event Calendar

Brief.

5.2.4 UC04: Create Event

ID: UC04 (Create Event) **Scope**: CS Automated Information Timeline
Level: User Goal
Stakeholders and Interests:

- Faculty: A person that works for the university and is interested in gaining visibility of their post and/or event.
- Admin/Reviewer: A person that works for the university and approves and/or removes posts and/or events from the system.

Preconditions:

- Faculty has been identified and authenticated.

Postconditions:

- Event has been persisted.
- Event has been placed in the proposed state.
- An event notification has been persisted.

Main Success Scenario:

1. Faculty writes the event using the event creation tool.
   (a) The event title is written by the faculty user.
   (b) The event body is written by the faculty user.
2. Faculty reviews the event draft.
3. Faculty submits the event to the system.
   (a) An event id is generated by the system.
   (b) A timestamp is generated by the system.
   (c) The user id of the faculty user is attached to the event object.
   (d) The system assigns the status of the event to the proposed state.
4. The system generates a notification object.

(a) The event object is attached to the notification object.

5. The notification object is persisted.
6. The event object is persisted.
7. The system returns the persisted event object to the faculty user.

Extensions:

*.a. Anytime the system does not respond,

1. Faculty will notify the Admin/Reviewer.
2. Admin/Reviewer will restart the system.

5.a. If the notification is not persisted,

1. The system returns an error message to the faculty user.
2. The faculty user resubmits the event.

6.a. If the event is not persisted,

1. The system returns an error message to the faculty user.
2. Faculty reattempts the submission of the event.

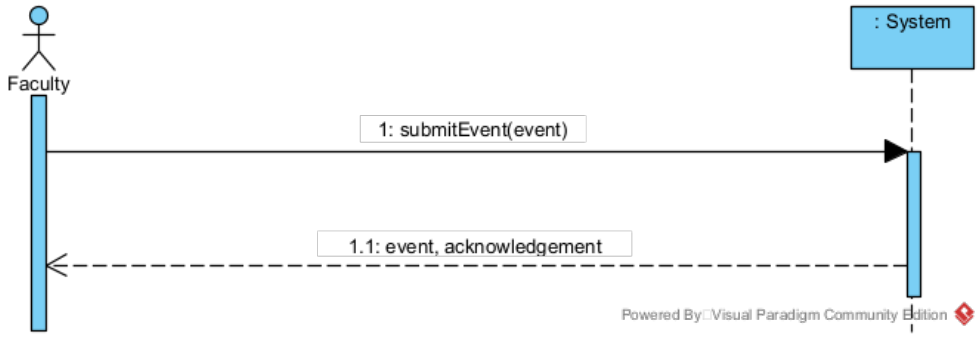

Figure 4: System Sequence Diagram: Create Event

Operation: submitEvent(event: Event)
Cross References: UC04 (Create Post)
Preconditions:

- Faculty, f, has been identified and authenticated.

Postconditions:

- Event, e, was created.
- Event e.status was set to “proposed”.
- Event, e, was persisted.

5.2.5 UC05: Edit Event

ID: UC05 (Edit Event)
Scope: CS Automated Information Timeline
Level: User Goal
Primary Actor: Faculty
Stakeholders and Interests:

- Audience: Wants to view up-to-date information about events in the CS Department
- Faculty: Wants the ability to provide updated information on events they submitted
- Office Manager: Wants to display accurate event information on the calendar on the Lobby TV
- Admin/Reviewer: Wants to update previously approved events with up-to-date information provided by the Faculty who submitted those events

Preconditions: Faculty created an event (UC04) and submitted it for review (UC08). Admin/Reviewer reviewed the event (UC09) and approved. Admin/Reviewer is identified and authenticated in the system. Faculty is identified and authenticated in the system and is viewing the event calendar (UC03).
Postconditions: Event status is in a proposed state.
Main Success Scenario:

1. Faculty clicks the "View My Events" link on the event calendar page.
2. System returns a list of existing events authored by Faculty that have been approved.
3. Faculty selects the event they wish to edit from the returned list of their previously approved events.
4. System returns the event page for the selected event.
5. Faculty clicks the "Edit This Event" button on the event page.
6. Faculty enters the updated event information into the submission form:
    - Event date
    - Event time
    - Event description
7. Faculty submits the updated event information for Admin/Reviewer review.
8. System provides Faculty with a submission confirmation message.

Extensions (or Alternate Flows):
3-4a. Faculty begins editing the event, then decides not to submit the updates for review:

1. Upon leaving the web page, any changes to event date, time, or description entered into the submission form are erased.
2. Selected event remains unchanged in the system.

3b. Faculty does not see the event they wish to edit in the returned list because the event has not yet been approved by the Admin/Reviewer:

1. Faculty must wait for approval from Admin/Reviewer on the original event before the event is eligible for editing.

Special Requirements: None
Technology and Data Variations List:
4a. Date input is required in the format MM/DD/YYYY.
4b. Time input is required in the Standard Time format in the Central Time Zone.
Frequency of Occurrence: Could be nearly continuous if Admin/Reviewer also reviews edited events and approves them (UC09) nearly continuously.
Open Issues: None

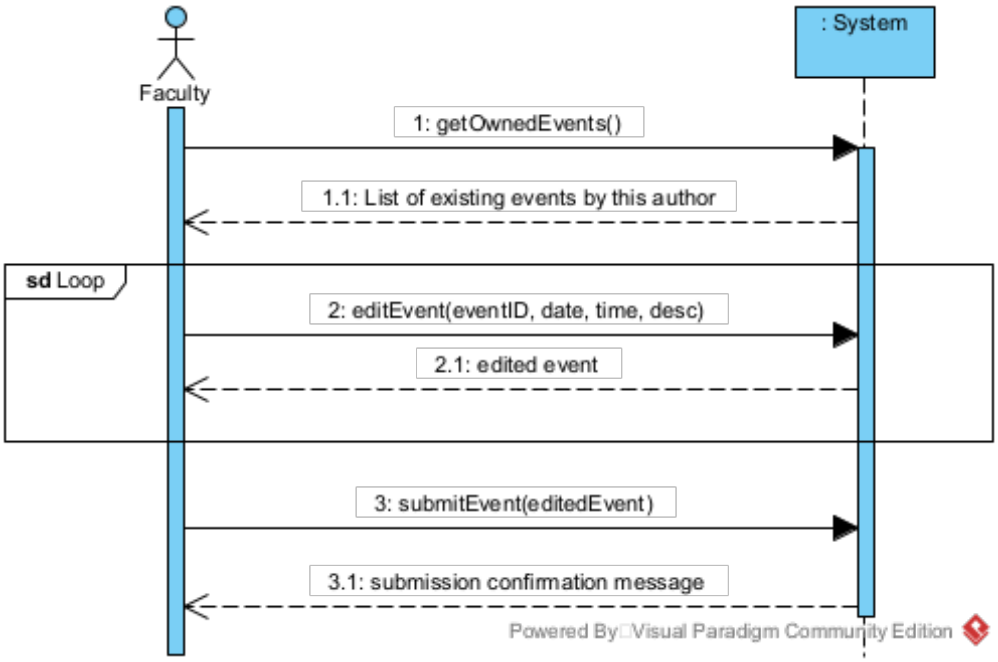

Figure 5: System Sequence Diagram: Edit Event

Operation: getOwnedEvents()
Cross-References: UC05 (Edit Event), UC06 (Delete Event), UC07 (View Event)
Pre-conditions: Event object has been created. Event has been associated with Faculty. Event attribute status is set to approved.
Post-conditions: None.

Operation: editEvent(eventID, date, time, desc)
Cross-References: UC05 (Edit Event)
Pre-conditions:

- Event object has been created.
- Event attribute status is set to approved.
- Event has been associated with Faculty.

Post-conditions: editedEvent has been created.

- editedEvent object has been created.
- editedEvent attributes (date, time, desc) have been updated.
- editedEvent has been associated with Faculty.

Operation: submitEvent(editedEvent)
Cross-References: UC05 (Edit Event)
Pre-conditions:

- editedEvent object has been created.
- editedEvent attributes (date, time, desc) have been updated.
- editedEvent has been associated with Faculty.

Post-conditions:

- Notification object is created.
- Notification object is associated with editedEvent, Faculty, and Admin/Reviewer.

5.2.6 UC06: Delete Event

ID: UC06 (Delete Event)
Scope: CS Automated Information Timeline
Level: User Goal
Primary Actor: Faculty
Stakeholders and Interests:

- Audience: Wants to view up-to-date information about events in the CS Department
- Faculty: Wants the ability to delete events that they submitted that are no longer happening
- Office Manager: Wants to display accurate event information on the calendar on the Lobby TV

Preconditions: Faculty created an event (UC04) and submitted it for review (UC08). Admin/Reviewer reviewed the event (UC09) and approved. Faculty is identified and authenticated in the system and is viewing the event calendar (UC03).
Postconditions: Event status is in a deleted state.
Main Success Scenario:

1. Faculty clicks the "View My Events" link on the event calendar page.
2. System returns a list of existing events authored by Faculty that have been approved.
3. Faculty selects the event they wish to delete from the returned list of their previously approved events.
4. System returns the event page for the selected event.
5. Faculty clicks the "Delete This Event" button on the event page.

6. System prompts Faculty to confirm the removal of the event from the event calendar.
7. Faculty confirms the removal of the event from the event calendar by clicking “Yes”.
8. System provides Faculty with a confirmation message.

Extensions (or Alternate Flows):
3a. Faculty does not see the event they wish to delete in the returned list because the event has not yet been approved by the Admin/Reviewer:

1. Faculty must wait for approval from Admin/Reviewer on the original event before the event is eligible for removal from the event calendar.

7a. Faculty decides they do not want to remove the event from the event calendar:

1. Faculty clicks the “No” button and is redirected to the list of existing events authored by Faculty that have been approved.

Special Requirements: None
Technology and Data Variations List: None
Frequency of Occurrence: Maximum once per submitted and approved event.
Open Issues: None

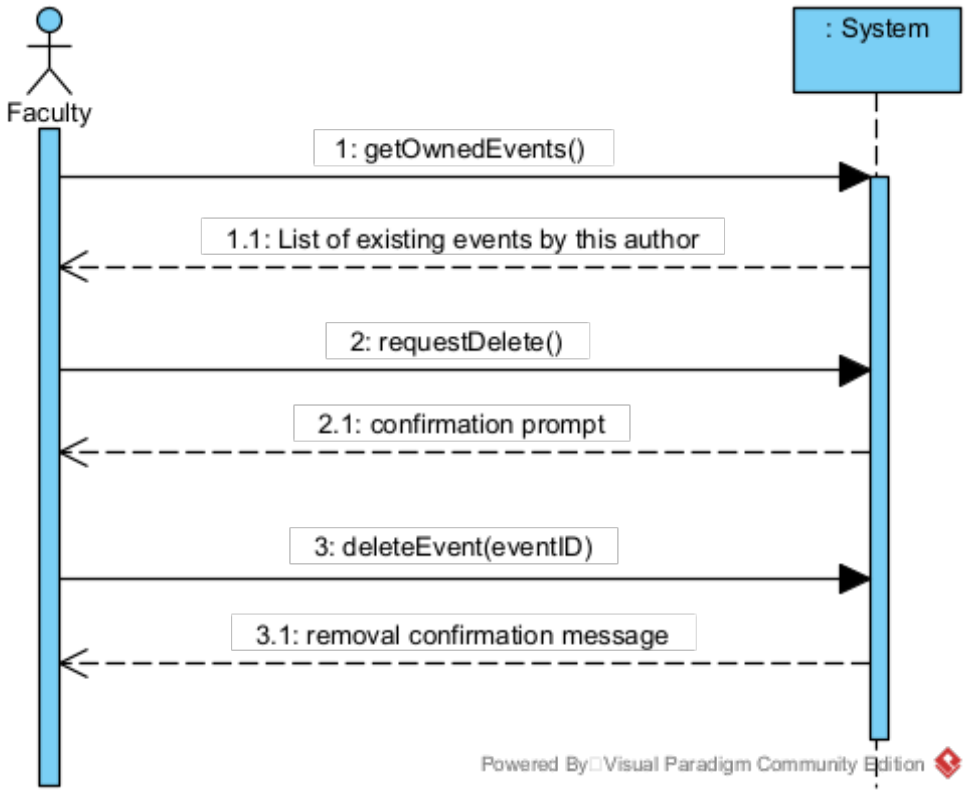

Figure 6: System Sequence Diagram: Delete Event

Operation: requestDelete()
Cross-References: UC06 (Delete Event)
Pre-conditions:

- Event object has been created.
- Event attribute status is set to approved.
- Event has been associated with Faculty.

Post-conditions: None

Operation: deleteEvent(eventID)
Cross-References: UC06 (Delete Event)
Pre-conditions:

- Event object has been created.
- Event attribute status is set to approved.
- Event has been associated with Faculty.

Post-conditions:

- Event attribute status is set to deleted.
- Event associated with Faculty has been broken.

5.2.7 UC07: View Event

ID: UC07 (View Event)
Scope: CS Automated Information Timeline
Level: User Goal
Primary Actor: Faculty
Stakeholders and Interests:

- Audience: Wants to view up-to-date information about events in the CS Department
- Faculty: Wants the ability to view events they have submitted
- Office Manager: Wants to display calendar events on the Lobby TV
- Admin/Reviewer: Wants to view events submitted by Faculty so they can review them

Preconditions: Faculty created an event (UC04) and submitted it for review (UC08). Admin/Reviewer reviewed the event (UC09) and approved. Faculty is identified and authenticated in the system and is viewing the event calendar (UC03).
Postconditions: Event status is in an approved state.
Main Success Scenario:

1. Faculty clicks the "View My Events" link on the event calendar page.

2. System returns a list of existing events authored by Faculty that have been approved.
3. Faculty selects the event they wish to view from the returned list of their previously approved events.
4. System returns the event page for the selected event.

Extensions (or Alternate Flows):
3a. Faculty does not see the event they wish to view in the returned list because the event has not yet been approved by the Admin/Reviewer:

1. Faculty must wait for approval from Admin/Reviewer on the original event before the event is eligible for viewing from the event calendar.

Special Requirements: None
Technology and Data Variations List: None
Frequency of Occurrence: Continuous
Open Issues: None

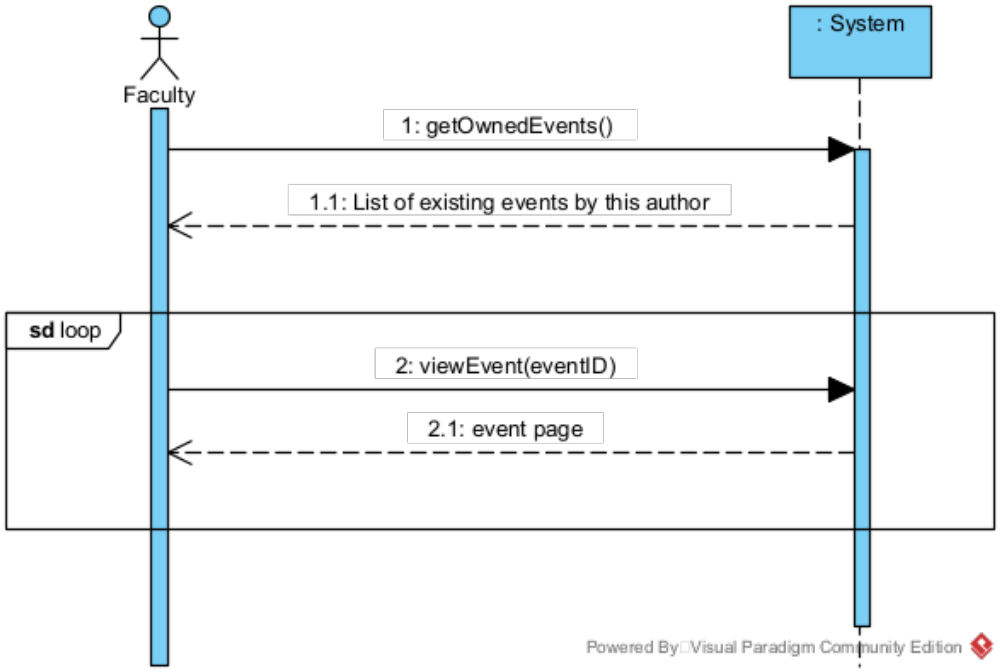

Figure 7: System Sequence Diagram: View Event

Operation: viewEvent(eventID)
Cross-References: UC07 (View Event)
Pre-conditions:

- Event object has been created.
- Event attribute status is set to approved.
- Event has been associated with Faculty.

Post-conditions: None

5.2.8 UC08: Submit For Review

Main Success Scenario: A faculty user submits a post, event, media, or page to the system. The system generates an id and a timestamp that is set to the system time upon submission and associates these attributes with the post, event, media, or page object. The system associates the faculty user that submitted with the post, event, media, or page. The system assigns the status of the post, event, media, or page to the proposed status. The system generates a notification object and attaches the post, event, media, or page object with the notification. The system persists both the created post, event, media, or page object and the created notification object to the database. The created post, event, media, or page is returned to the faculty user.

Alternative Flows:

- The post, event, media, page, or notification object is not persisted.
  - The system will respond to the faculty user with an error message indicating the reason for persistence failure.
  - The faculty member will correct the identified error and attempt to resubmit the post, event, media, or page item.
- If the system does not respond within a given timeframe.
  - The system will return an error message to the faculty user indicating a timeout has occurred.
  - The faculty user will reattempt the submission of the post, event, media, or page item.
    - If the system continues to fail to respond within a given timeframe.
      - The faculty user will inform the admin/reviewer of the error.
      - The admin/reviewer will restart the system.
      - The faculty user will reattempt submission of the post, event, media, or page item.

Figure 8: System Sequence Diagram: Submit For Review

5.2.9 UC09: Review For Approval

ID: UC09 (Review for Approval)
Scope: CS Automated Information Timeline

Level: User Goal
Stakeholders and Interests:

- Faculty: A person that works for the university and is interested in gaining visibility of their post and/or event.
- Administrator/Reviewer: A person that works for the university and is interested in reviewing and approving requests for system content additions.

Preconditions:

- Administrator/Reviewer a has been identified and authenticated.
- Notification n has been created and persisted.
- n.event, n.post, or n.media has been set.

Postconditions:

- Notification n.entity was identified as either a Post, Event, or Media.
- n.entity.status was set to either "Approved" or "Rejected".
- If the review was favorable, n.entity.status was set to "Approved".
- If the review was unfavorable, n.entity.status was set to "Rejected".
- n.reviewer was set to a.
- n.entity was persisted.

Main Success Scenario:

1. Administrator/Reviewer, a, gets all notifications.
2. For each Notification, n, in notifications, Administrator/Reviewer, a, performs manual review of the Post, Event, or Media attached to Notification, n.
3. Administrator/Reviewer, a, updates the Post, Event, or Media that was attached to Notification, n, to either "Approved" or "Rejected".
4. If the review is favorable, n.event.status, n.post.status, or n.media.status is set to "Approved".
5. If the review is not favorable, n.event.status, n.post.status, or n.media.status is set to "Rejected".
6. The system persists the updated Post, Event, or Media with the status set by Administrator/Reviewer, a.
7. The system associates Administrator/Reviewer, a, with Notification n.
8. Notification, n, is persisted.
9. The Event, Post, or Media is persisted.

Extensions:
2a. There are multiple objects, Post, Event, or Media, attached to Notification n.

- Administrator/Reviewer, a, deletes Notification, n.

Special Requirements: None
Technology and Data Variation List:

- Notification n.post is set and of Post type; n.event and n.media are null.
- Notification n.event is set and of Event type; n.post and n.media are null.
- Notification n.media is set and of Media type; n.post and n.event are null.

Frequency of Occurrence: Could be nearly continuous.

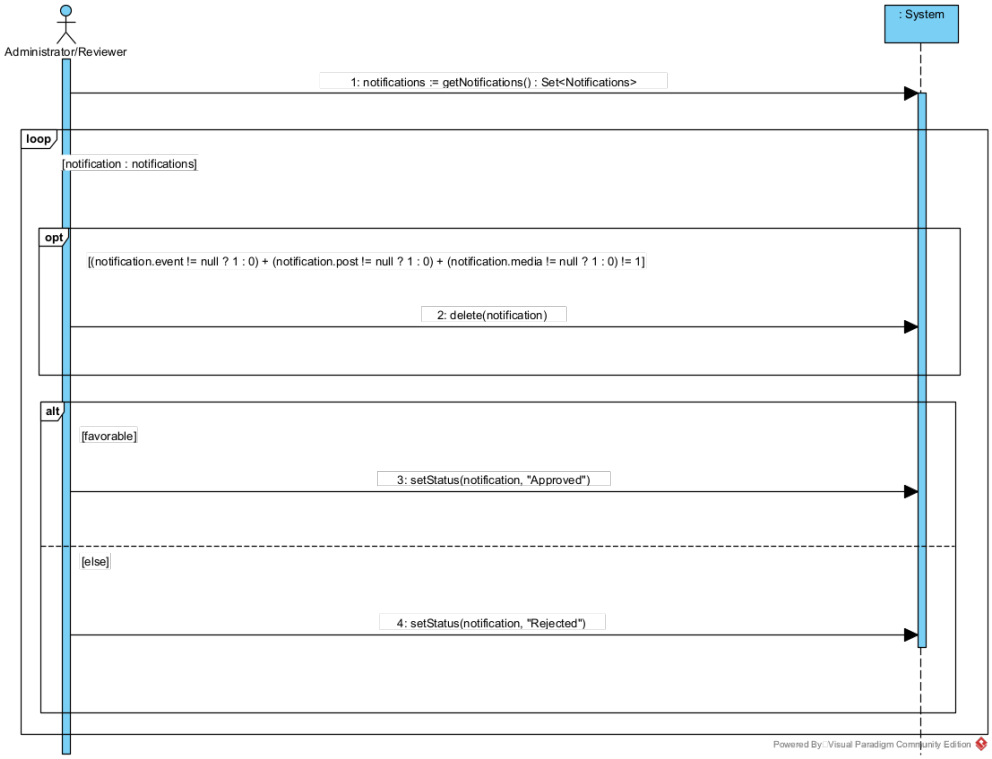

Figure 9: System Sequence Diagram: Review For Approval

Operation: getNotifications()
Cross References: UC09 (Review for Approval)
Preconditions:

- Admin/Reviewer, a, has been identified and authenticated.
- Notification, n, has been created and persisted.
- Notification n.entity has been set to Post p, Event e, and/or Media m.

Postconditions: None

Operation: deleteNotification(notification: Notification)
Cross References: UC09 (Review for Approval)
Preconditions:

- Admin/Reviewer, a, has been identified and authenticated.
- Notification, n, has been created and persisted.
- Notification n.entity has been set to Post p, Event e, and/or Media m.

Postconditions:

- Association n.entity was disassociated.
- All entities, Post p, Event e, and/or Media m, from association n.entity were removed from persistence and destroyed.
- Notification, n, was removed from persistence and destroyed.

Operation: setStatus(status: String)
Cross References: UC09 (Review for Approval)
Preconditions:

- Admin/Reviewer, a, has been identified and authenticated.
- Notification, n, has been created and persisted.
- n.reviewer has been set to Admin/Reviewer a.
- n.entity has been set to one of Post p, Event e, or Media m.

Postconditions:

- Notification n.entity.status was set to status.
- Notification n.reviewer was set to a.
- Notification, n, was persisted.

5.2.10 UC10: Create Post

ID: UC10 (Create Post)
Scope: CS Automated Information Timeline
Level: User Goal
Stakeholders and Interests:

- Faculty: A person who works for the university and is interested in gaining visibility of this post and/or event.

- Admin/Reviewer: A person who works for the university and approves and/or removes posts and/or event from the system.

Preconditions:

- Faculty, f, has been identified and authenticated.

Postconditions:

- Post, p, was created.
- p.status was set to "Pending".
- p.createdBy was set to Faculty f.
- Post, p, was persisted.
- Notification, n, was created.
- n.post was set to Post p.
- Notification, n, was persisted.

Main Success Scenario:

1. Faculty writes the post using the post creation tool.
2. The post title is written by the faculty user.
3. The post body is written by the faculty user.
4. Faculty reviews the post draft.
5. Faculty submits the post to the system.
6. A post id is generated by the system.
7. A timestamp is generated by the system.
8. The user id of the faculty user is attached to the post object.
9. The system assigns the status of the post to the proposed state.
10. The system generates a notification object.
11. The post object is attached to the notification object.
12. The post object is persisted.
13. The notification object is persisted.
14. The system returns the persisted post object to the faculty user.

Extensions:
*.a. Anytime the system does not respond.

1. Faculty will notify the Admin/Reviewer.

2. Admin/Reviewer will restart the system.
3. Faculty will recreate and submit the post.

5.a. Anytime the system does not respond.

1. The system will return an error message to the faculty user.
2. The faculty user will resubmit the post.

6.a. If the post is not persisted.

1. The system will return an error message to the faculty user.
2. The faculty user will reattempt the submission of the post.

Special Requirements: None
Technology and Data Variations List:

1. Date will be of the format "yyyy-MM-ddTHH:mm:ss".
2. Id will be of the format of a universally unique identifier, UUID.

Frequency of Occurrence: Could be nearly continuous.

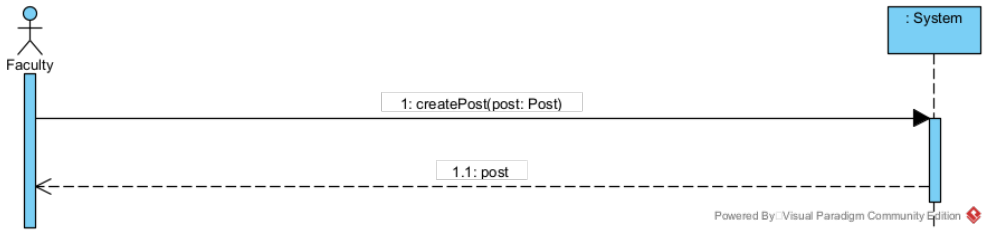

Figure 10: System Sequence Diagram: Create Post

Operation: createPost(post: Post)
Cross References: UC10 (Create Post)
Preconditions:

- Faculty, f, has been identified and authenticated.

Postconditions:

- Post, p, was created.
- p.status was set to "Pending".
- p.createdBy was set to Faculty f.
- Post, p, was persisted.
- Notification, n, was created.
- n.post was set to Post p.
- Notification, n, was persisted.

5.2.11 UC11: Edit Post

ID: UC11 (Edit Post)
Scope: CS Automated Information Timeline
Level: User goal
Primary Actor: Faculty or Admin/Reviewer
Stakeholders and Interests:

- Audience: A person that is interested in viewing all approved content on the system using their mobile device.
- Faculty: A person that works for the university and is interested in gaining visibility of their post and/or event.
- Office Manager: A person that works for the university and is interested in prioritizing the order of posts and/or events.
- Admin/Reviewer: A person that works for the university and approves and/or removes posts and/or events from the system.

Preconditions:

- Faculty has been identified and authenticated.
- A post has been created.

Postconditions:

- Post has been updated and saved to database.
- Media library has been updated with new images and videos from the updated post.
- The display board reflects the updated post flagged for display.

Main Success Scenario:

1. User selects a post to edit
2. System opens the editable view for the selected post with options to edit the text, delete the photos and videos, if present, on the post, upload photos and videos on the post and update the display flag.
3. User saves the updated post.
4. System presents the confirmation that the post is updated

Alternative Flows:
a. At any time, system fails or becomes unresponsive and does not provide an error message

1. User performs a hard refresh on the browser (ctrl + f5 or shift + reload)
2. System reloads the editable view for the post

b. (3.a) Uploaded media is in unsupported format or exceeds the file size limit

1. System prompts the user with the appropriate error message
2. User acknowledges the error and reuploads the media in supported format and size.

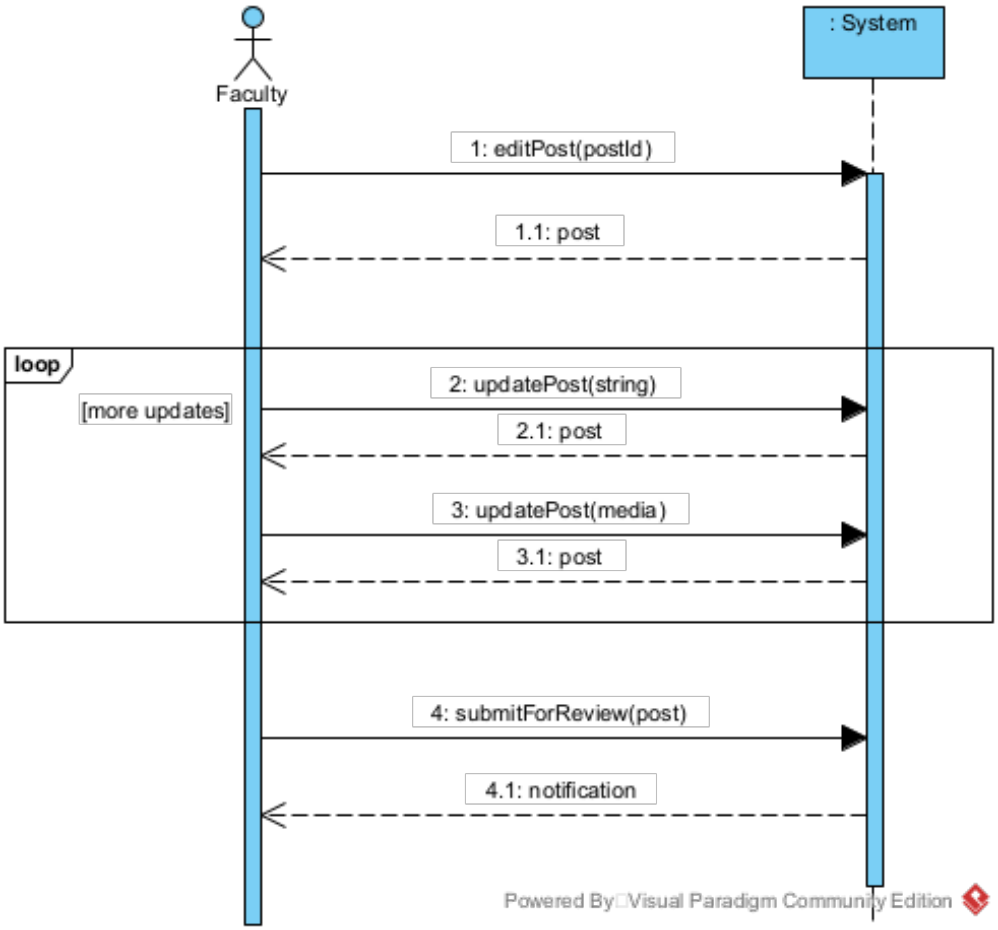

Figure 11: System Sequence Diagram: Edit Post

Operation: updatePost(post)
Cross Reference: UC11 (Edit Post)
Preconditions:

- Faculty has been identified and authenticated.
- A post has been created and persisted in the database.
- Update operation is underway.

Postconditions:

- A Post instance, post was created
- post was associated with the Update operation.

- post was modified
- Modified post was persisted to the database

5.2.12 UC12: Delete Post

ID: UC12 (Delete Post)
Scope: CS Automated Information Timeline
Level: User goal
Primary Actor: Faculty or Admin/Reviewer
Stakeholders and Interests:

- Audience: A person that is interested in viewing all approved content on the system using their mobile device.
- Faculty: A person that works for the university and is interested in gaining visibility of their post and/or event.
- Office Manager: A person that works for the university and is interested in prioritizing the order of posts and/or events.
- Admin/Reviewer: A person that works for the university and approves and/or removes posts and/or events from the system.

Preconditions:

- Faculty has been identified and authenticated.
- A post to be deleted exists.

Postconditions:

- The post has been deleted from the database.
- Media library has been updated.
- The display board no longer displays the deleted post.

Main Success Scenario:

1. User navigates to the admin dashboard and accesses the manage post view.
2. User selects the post to be managed and clicks the delete post button.
3. System presents the confirmation window prompting the user if they are sure about deleting the post
4. User confirms deletion.
5. The system deletes the selected post from the database and updates the display board contents.
6. System presents the confirmation that the post is deleted

Alternative Flows:
a. At any time, system fails or becomes unresponsive and does not provide an error message

1. User performs a hard refresh on the browser (ctrl + f5 or shift + reload)
2. System reloads the editable view for the post

b. (4.a) User does not confirm post deletion (clicks no)

1. No action is taken, and the post is displayed in manage post view

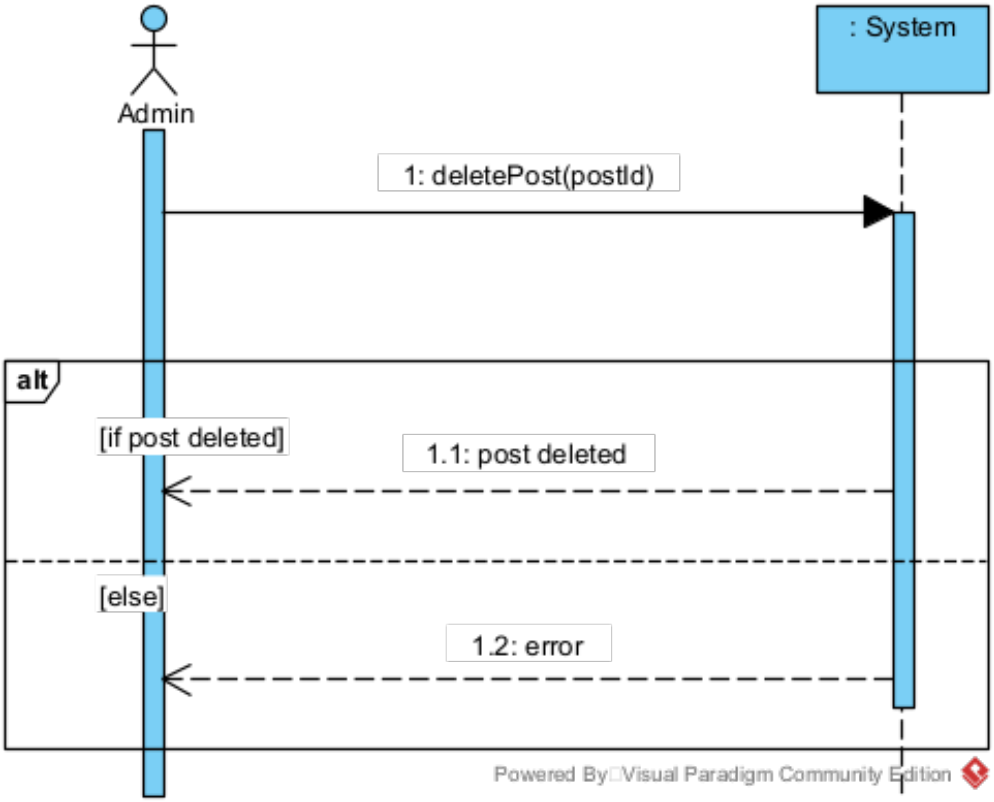

Figure 12: System Sequence Diagram: Delete Post

Operation: deletePost(postId)
Cross Reference: UC12 (Delete Post)
Preconditions:

- Faculty has been identified and authenticated.
- A post has been created and persisted in the database.
- Delete operation is underway.

Postconditions:

- A Post instance, post was created
- post was associated with the Delete operation.
- post was deleted

5.2.13 UC13: View Post

ID: UC13 (View Post)
Scope: CS Automated Information Timeline
Level: User goal
Primary Actor: Audience
Stakeholders and Interests:

- Audience: A person that is interested in viewing all approved content on the system using their mobile device.
- Faculty: A person that works for the university and is interested in gaining visibility of their post and/or event.
- Office Manager: A person that works for the university and is interested in prioritizing the order of posts and/or events.
- Admin/Reviewer: A person that works for the university and approves and/or removes posts and/or events from the system.

Preconditions:

- A post has been created and staged for display.

Postconditions: None
Main Success Scenario:

1. User accesses the web application UI by scanning the QR code on the display board.
2. System presents the web view with posts staged for display

Alternative Flows:
a. At any time, system fails or becomes unresponsive and does not provide an error message

1. User performs a hard refresh on the browser (ctrl + f5 or shift + reload)
2. System reloads the editable view for the post

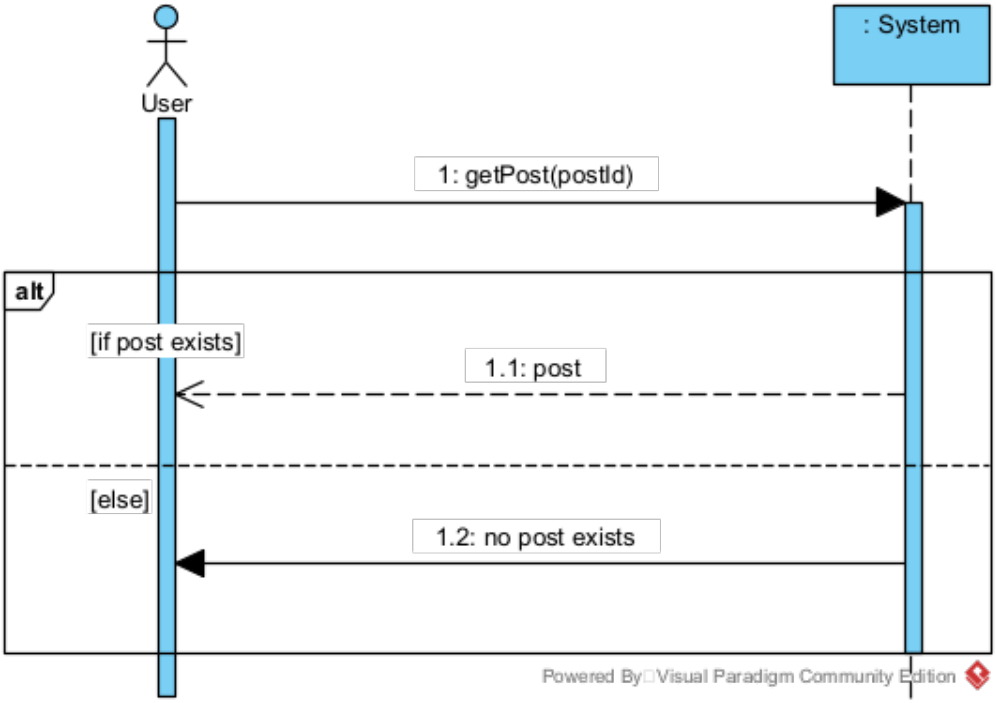

Figure 13: System Sequence Diagram: View Post

Operation: getPost(postId)
Cross Reference: UC13 (View Post) **Preconditions:**

- Post exists and is staged for display
- User is attempting to view a specific post (request)

Postconditions:

- A PostService instance, postService was created
- A PostRepository instance postRepository was created
- A Post instance post was created
- post was associated with the request

5.2.14 UC14: Update HTML on Page

Brief.

5.2.15 UC15: Manage Displayed Posts

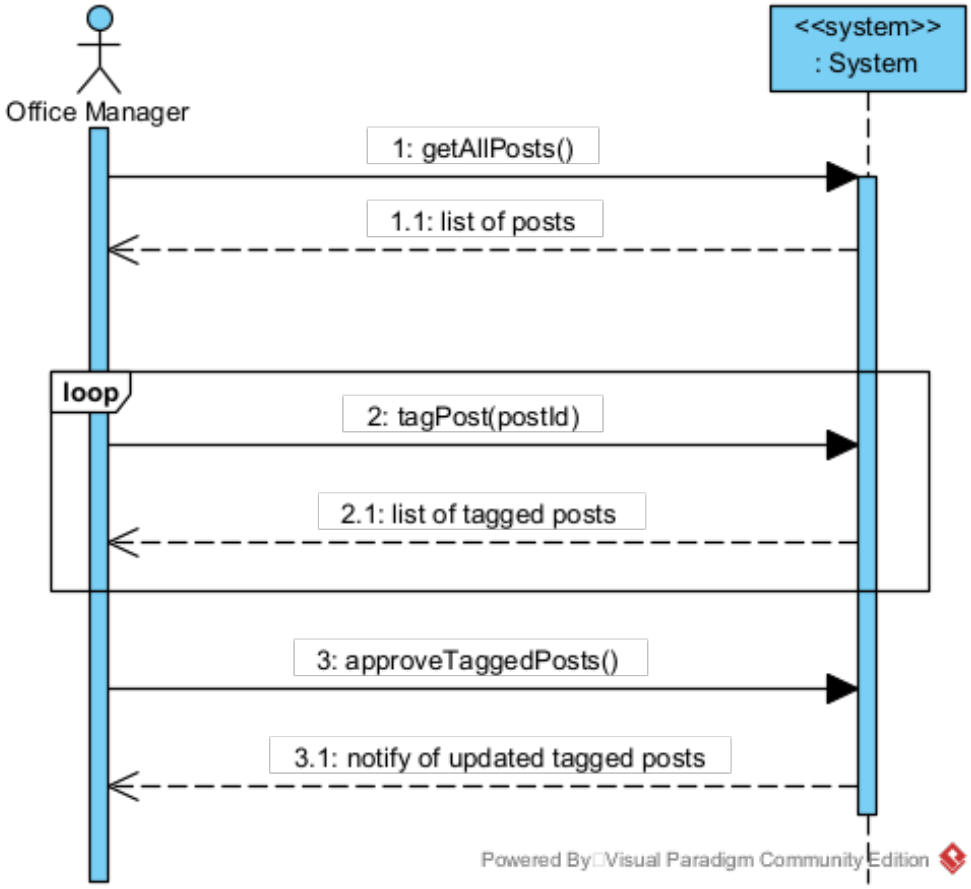

Figure 14: System Sequence Diagram: Manage Displayed Posts

5.2.16 UC16: Manage Displayed Media

ID: UC16 (Manage Media For Display)
Scope: CS Automated Information Timeline
Level: User goal
Primary Actor: Office Manager
Stakeholders and Interests:

- Admin/Reviewer: Wants to ensure curated list of media items are on display for guests
- Guest: Wants to be able to see images and/or video on the TV display
- Faculty: Wants to ensure relevant media can be shown to guests.

Preconditions:

- User with Office Manager role has been authenticated.

- Media library is available.

Postconditions:

- Media is displayed on the TV display
- The ordering of the media displayed is correct

Main Success Scenario:

1. User navigates to media library
2. The system displays the media library and available files for display
3. User selects media to be displayed on the main display
4. The system displays the list of media that will be displayed on the main display
5. Repeat step 3 and 4 until all required media is selected for display
6. User reviews final list of media to be displayed
7. User approves media display and saves new list of media to be displayed
8. The system records the final list and the user identification associated with the created list
9. The system updates the main display with the list of media to display
10. The system provides local copies of media to the main display
11. The main display begins displaying the listed media
12. The system notifies the user that the listed media has been saved and is displayed on the main display

Alternative Flows:
3A: User does not find suitable media for display and has media to upload

1. User selects media they wish to upload from local device
2. User uploads new media to media library
3. The system confirms successful upload of media
4. The system adds newly uploaded media to list of media to display
5. Proceed to step 4 of the main scenario

3B: User wants to remove items from current Media list

1. User proceeds to modify currently returned media list
2. System returns the current list of media marked for display on the main display
3. User updates list to remove media
4. System returns updated list

5. Repeat step 3 and 4 until either all media is removed from the list or user cancels the action.
6. Proceed to step 6 of the main scenario

7A: User needs to modify order of media to be displayed

1. User selects option to change media ordering
2. The system notifies the user the current order of media will be lost
3. User confirms
4. The system returns user to step 3 of the main scenario

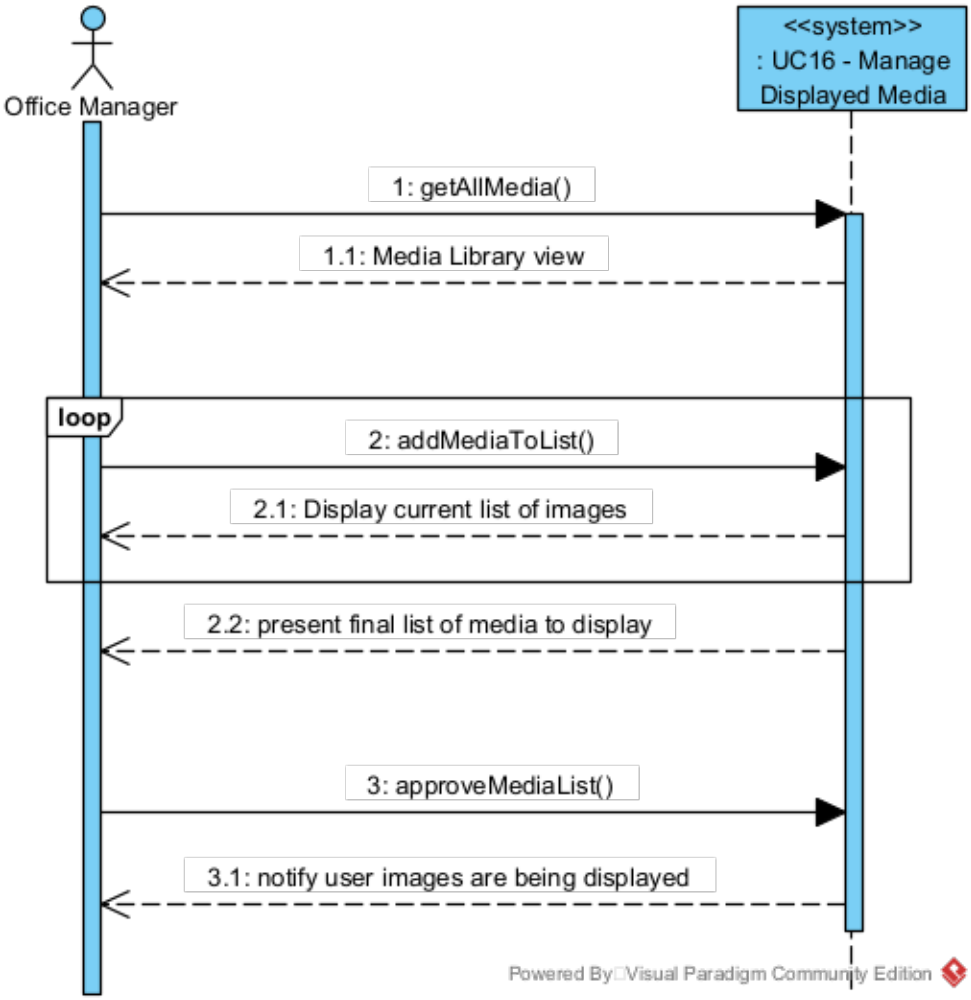

Figure 15: System Sequence Diagram: Manage Displayed Media

Operation: getAllMedia()
Cross-References: UC16 (Manage Media For Display)
Pre-conditions:

- Media objects exist in the media library (database store)
- Media objects have been approved by admin

Post-conditions:

- List of media returned to user as List¡¿ object, *ml*

Operation: addMediaToList()
Cross-References: UC16 (Manage Media For Display)
Pre-conditions:

- Media List object, *ml*, is not at capacity yet
- *ml* has not been approved

Post-conditions:

- *ml* has association with media item, *m*, formed
- *ml* has been updated and requests approval

Operation: approveMediaList ()
Cross-References: UC16 (Manage Media For Display)
Pre-conditions:

- User has marked *ml* as ready for publish
- *ml* is not over capactiy
- *ml* has not been approved yet

Post-conditions:

- *ml* is approved and set for display
- *MainDisplay* updates with new list, *ml*

6 Activity Diagram: Create Event

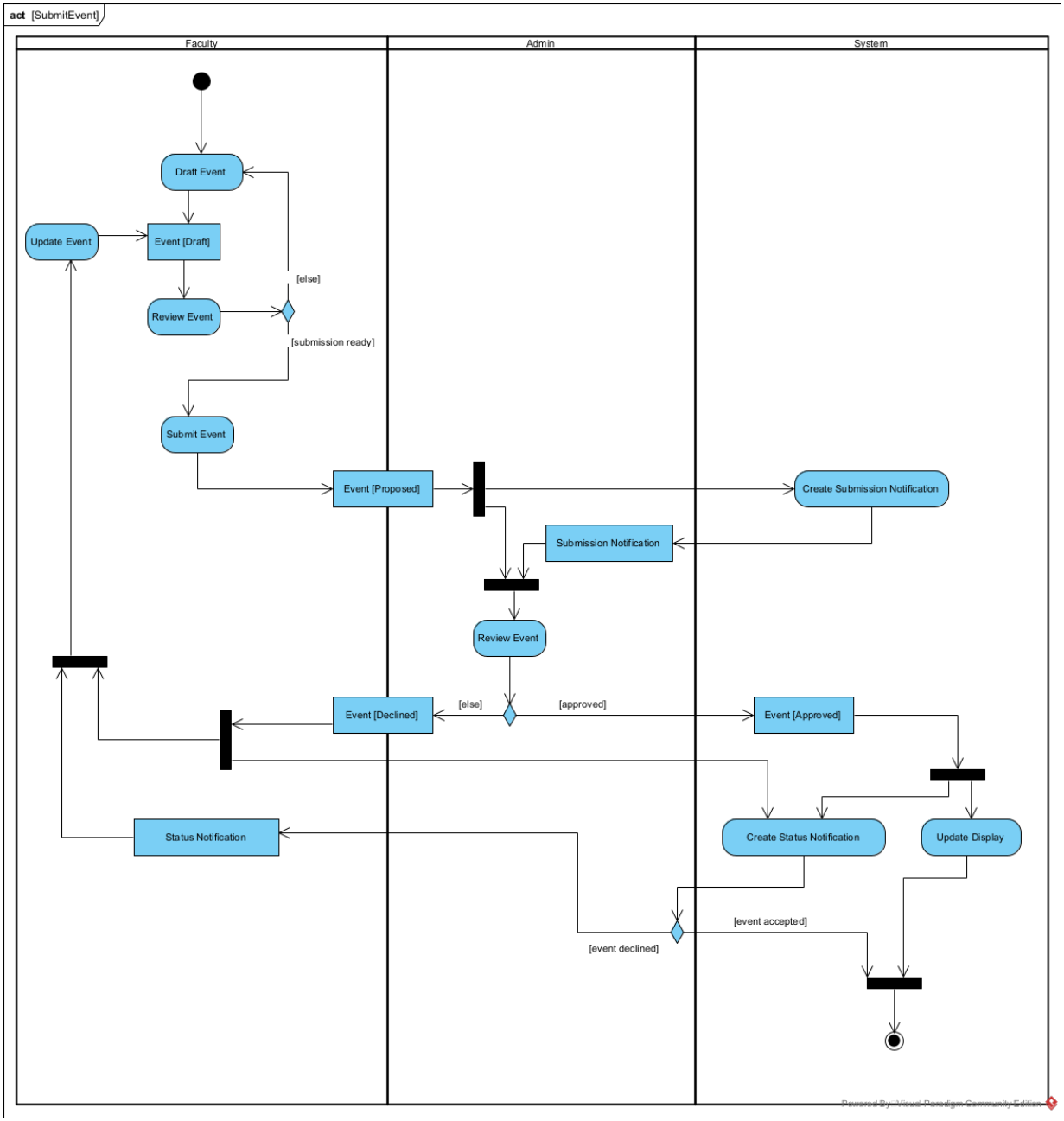

Figure 16: Activity Diagram: Create Event

7 Wireframes

7.1 TV

7.2 Website

8 Class Diagram

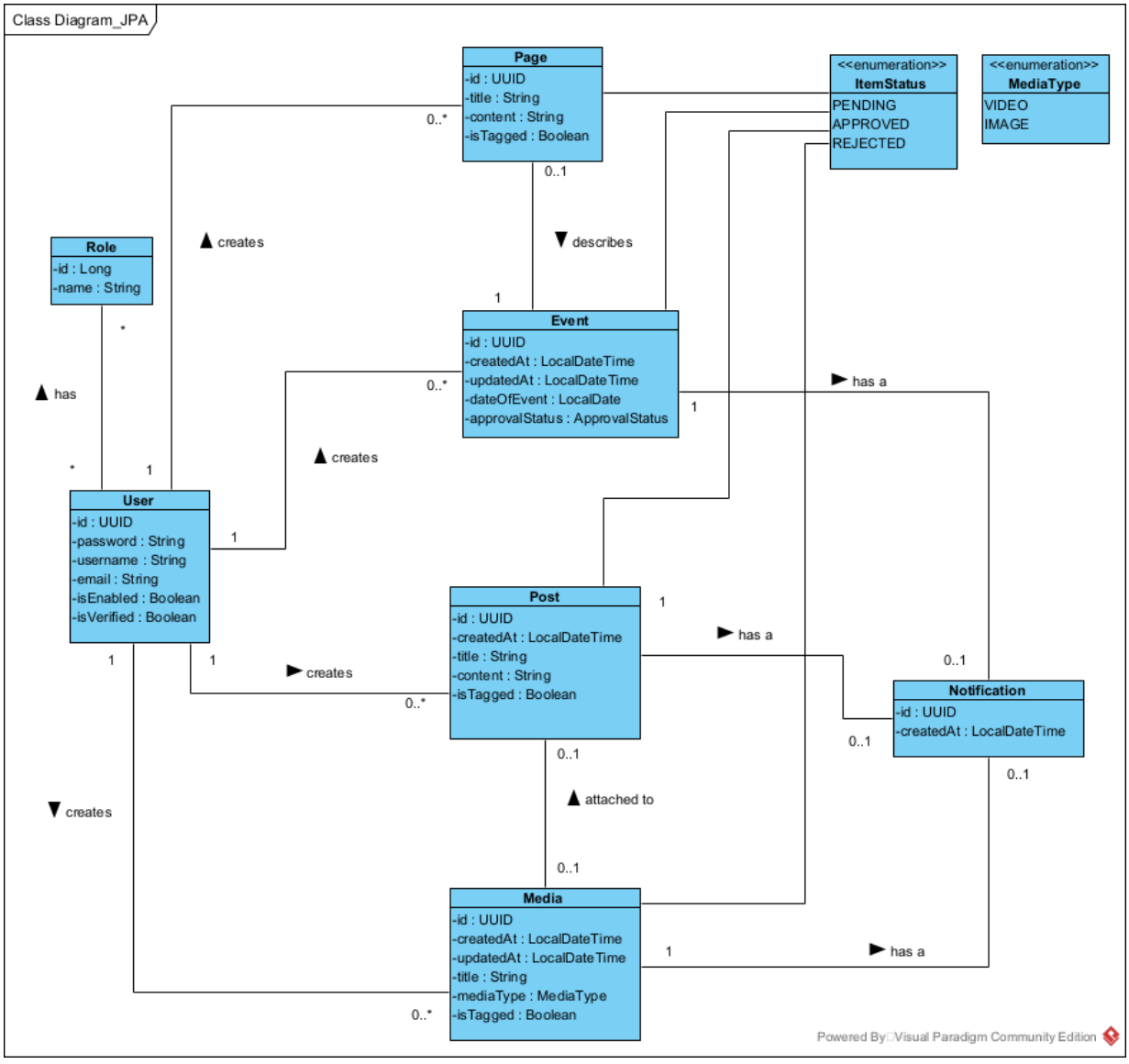

Figure 17: Class Diagram

9 Code

Note about the code.