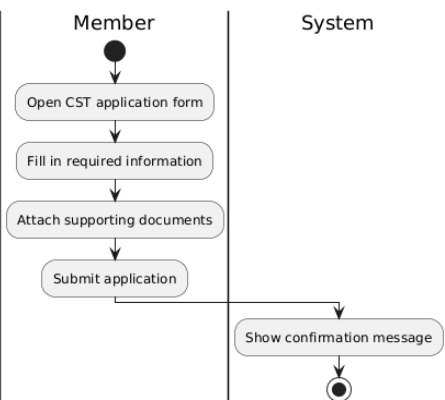
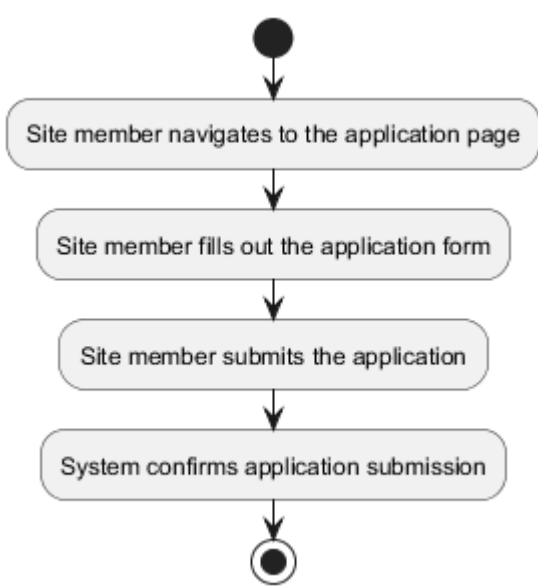
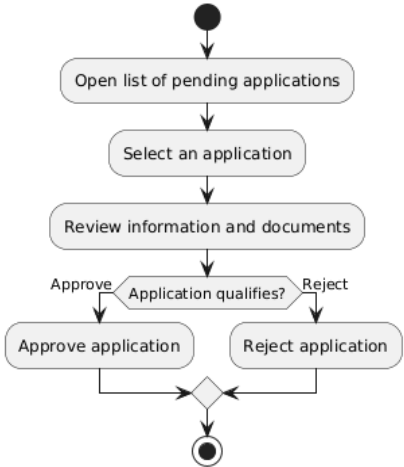
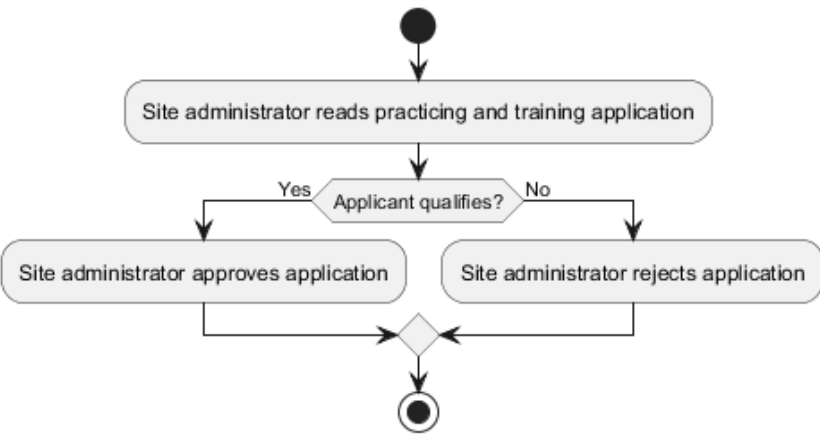
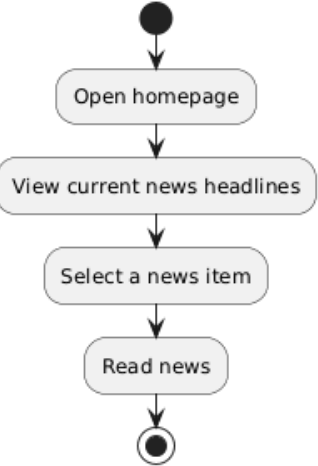
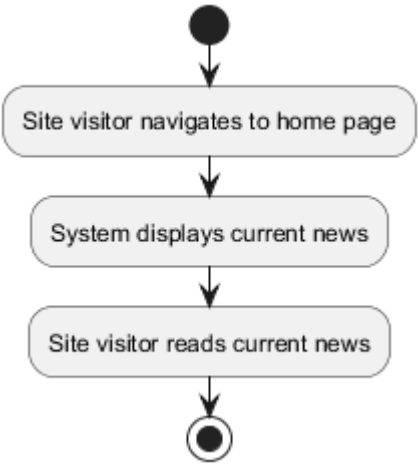
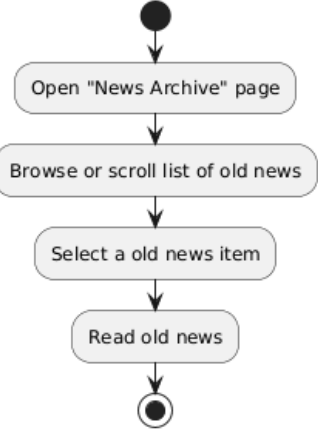
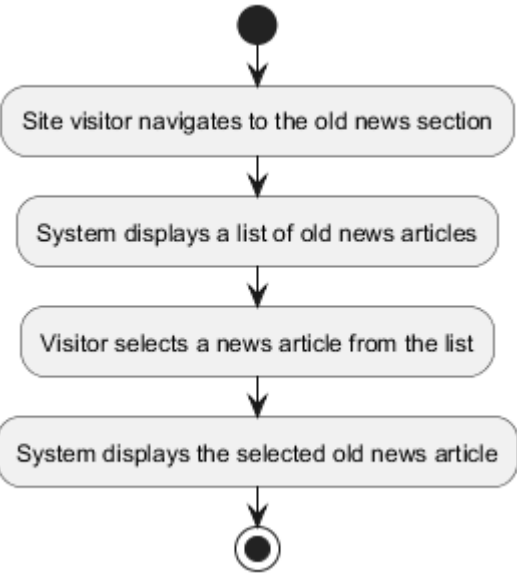
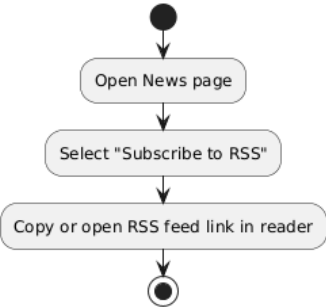
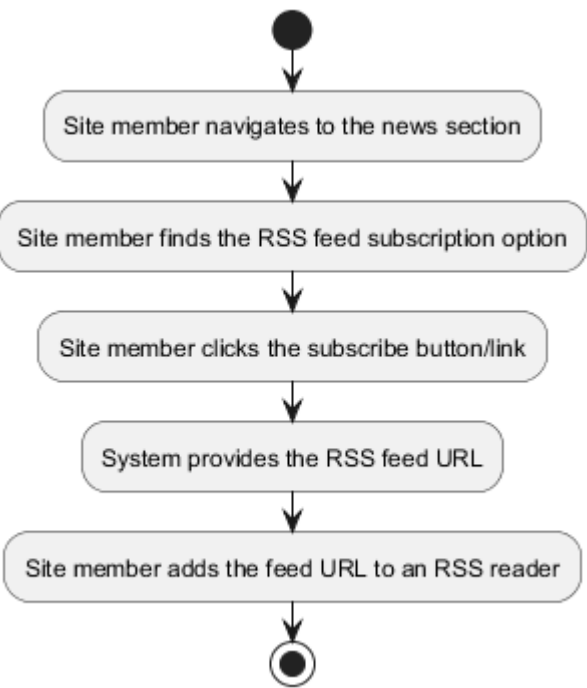


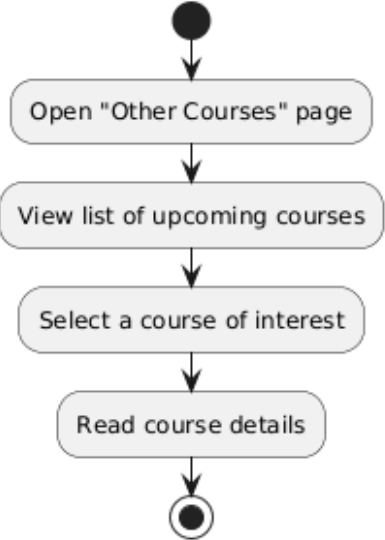
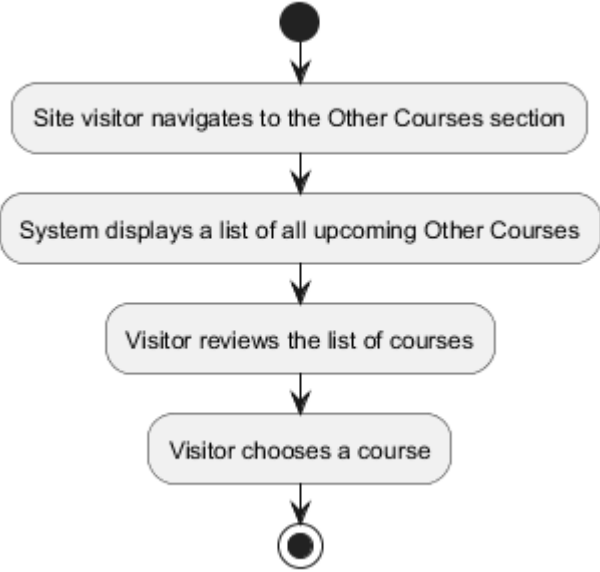
DIAGRAMAS DE REFERENCIA Y DIAGRAMAS GENERADOS

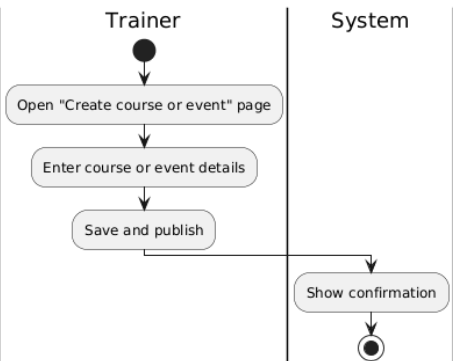
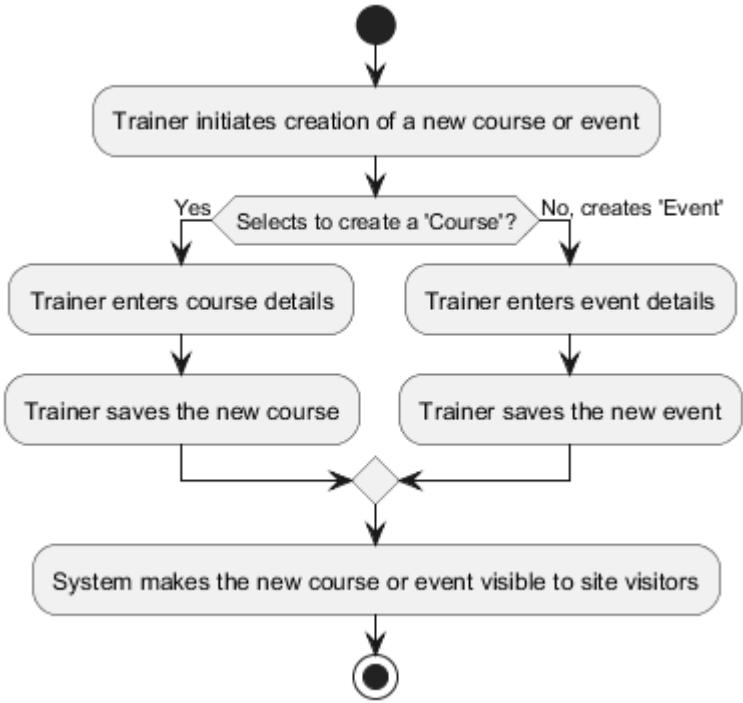
ID	Diagramas de Referencia (Humano)		Diagramas Generados (Herramienta)	
	Código en PlantUML	Diagrama en PlantUML	Código en PlantUML	Diagrama en PlantUML
HU-AC01	<pre>@startuml title Apply for Certified Scrum Practitioner (CSP) [Member] start :Open "CSP Application"; [System] :Display application form; [Member] :Fill required fields; :Attach evidence (files); :Preview application; :Submit; [System] if (Input valid & files ok?) then (Yes) :Persist application; :Assign trackingId; :Queue confirmation email; :Show success message; stop else (No) :Show inline errors; :Keep entered data; endif @enduml</pre>	<p>The diagram shows a Member actor and a System boundary. The Member starts with 'Open "CSP Application"', which triggers the System to 'Display application form'. The Member then performs 'Fill required fields', 'Attach evidence (files)', and 'Preview application' before 'Submit'. The System then checks 'Input valid & files ok?'. If 'Yes', it proceeds to 'Persist application', 'Assign trackingId', 'Queue confirmation email', and 'Show success message' before ending. If 'No', it shows 'Show inline errors' and 'Keep entered data' before returning to the Member.</p>	<pre>@startuml start :Site member accesses application form; :Site member fills out application; :Site member submits application; if (Application is approved?) then (Yes) :System grants Certified Scrum Practitioner designation; else (No) :System notifies member of rejection; endif stop @enduml</pre>	<p>The activity diagram starts with a start node leading to 'Site member accesses application form', followed by 'Site member fills out application' and 'Site member submits application'. A decision node 'Application is approved?' follows. The 'Yes' path leads to 'System grants Certified Scrum Practitioner designation', and the 'No' path leads to 'System notifies member of rejection'. Both paths merge at a join node before reaching the end node.</p>

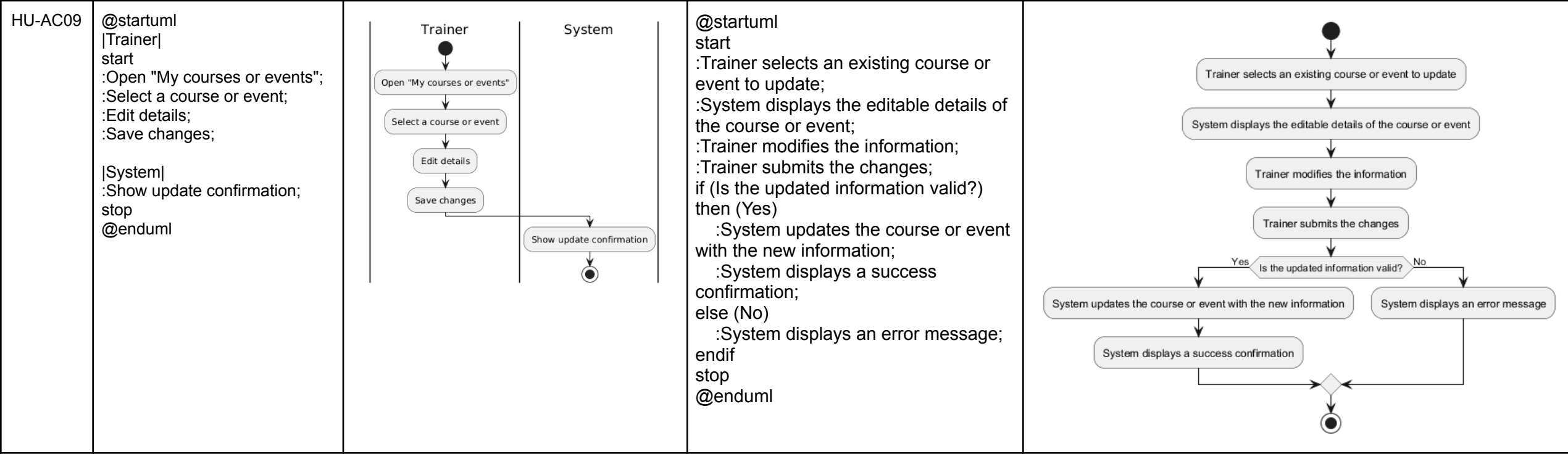
HU-AC02	<p>@startuml Member start :Open CST application form; :Fill in required information; :Attach supporting documents; :Submit application; System :Show confirmation message; stop @enduml</p>	 <pre>graph TD subgraph Member direction TB Start(()) --> U1[Open CST application form] U1 --> U2[Fill in required information] U2 --> U3[Attach supporting documents] U3 --> U4[Submit application] end subgraph System direction TB U4 --> U5[Show confirmation message] U5 --> End(()) end</pre>	<p>@startuml start :Site member navigates to the application page; :Site member fills out the application form; :Site member submits the application; :System confirms application submission; stop @enduml</p>	 <pre>graph TD Start(()) --> U1[Site member navigates to the application page] U1 --> U2[Site member fills out the application form] U2 --> U3[Site member submits the application] U3 --> U4[System confirms application submission] U4 --> End(())</pre>
HU-AC03	<p>@startuml start :Open list of pending applications; :Select an application; :Review information and documents; if (Application qualifies?) then (Approve) :Approve application; else (Reject) :Reject application; endif stop @enduml</p>	 <pre>graph TD Start(()) --> U1[Open list of pending applications] U1 --> U2[Select an application] U2 --> U3[Review information and documents] U3 --> D1{Application qualifies?} D1 -- Approve --> U4[Approve application] D1 -- Reject --> U5[Reject application] U4 --> J1{ } U5 --> J1 J1 --> End(())</pre>	<p>@startuml start :Site administrator reads practicing and training application; if (Applicant qualifies?) then (Yes) :Site administrator approves application; else (No) :Site administrator rejects application; endif stop @enduml</p>	 <pre>graph TD Start(()) --> U1[Site administrator reads practicing and training application] U1 --> D1{Applicant qualifies?} D1 -- Yes --> U2[Site administrator approves application] D1 -- No --> U3[Site administrator rejects application] U2 --> J1{ } U3 --> J1 J1 --> End(())</pre>

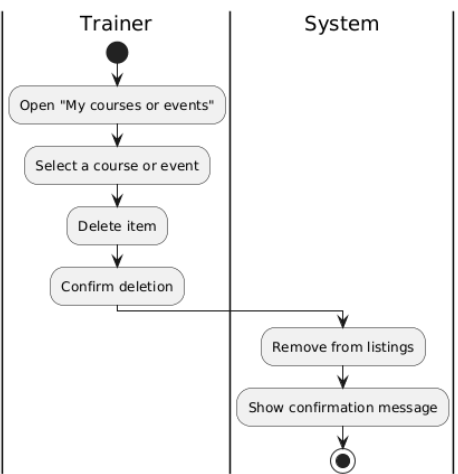
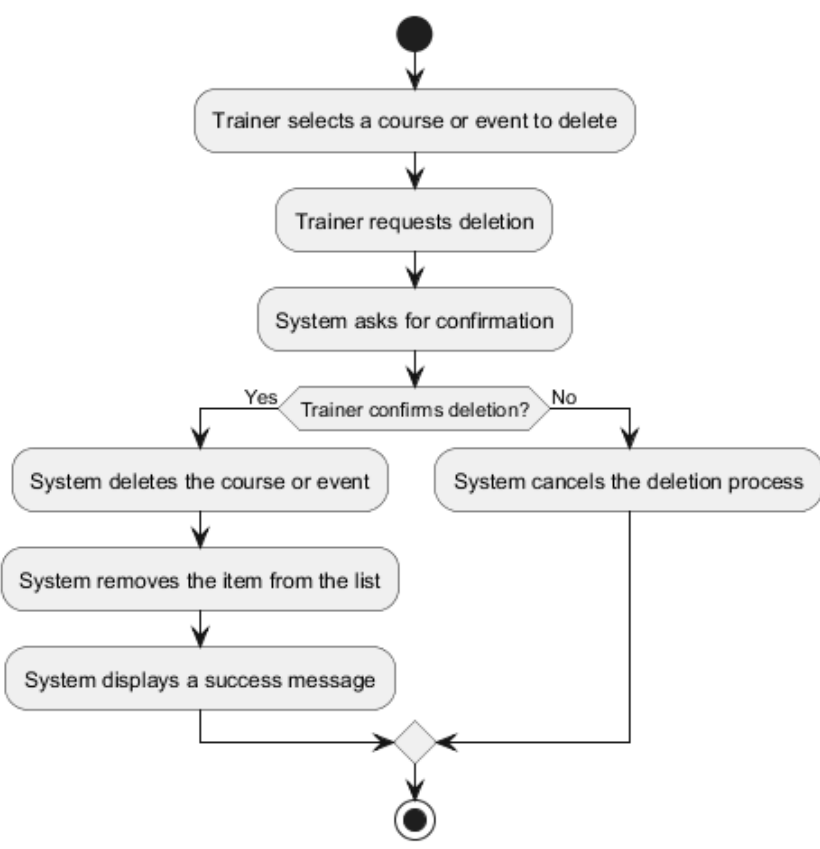
HU-AC04	<p>@startuml start :Open homepage; :View current news headlines; :Select a news item; :Read news; stop @enduml</p>	 <pre>graph TD; Start(()) --> UC1([Open homepage]); UC1 --> UC2([View current news headlines]); UC2 --> UC3([Select a news item]); UC3 --> UC4([Read news]); UC4 --> End((()))</pre>	<p>@startuml start :Site visitor navigates to home page; :System displays current news; :Site visitor reads current news; stop @enduml</p>	 <pre>graph TD; Start(()) --> UC1([Site visitor navigates to home page]); UC1 --> UC2([System displays current news]); UC2 --> UC3([Site visitor reads current news]); UC3 --> End((()))</pre>
HU-AC05	<p>@startuml start :Open "News Archive" page; :Browse or scroll list of old news; :Select a old news item; :Read old news; stop @enduml</p>	 <pre>graph TD; Start(()) --> UC1([Open "News Archive" page]); UC1 --> UC2([Browse or scroll list of old news]); UC2 --> UC3([Select a old news item]); UC3 --> UC4([Read old news]); UC4 --> End((()))</pre>	<p>@startuml start :Site visitor navigates to the old news section; :System displays a list of old news articles; :Visitor selects a news article from the list; :System displays the selected old news article; stop @enduml</p>	 <pre>graph TD; Start(()) --> UC1([Site visitor navigates to the old news section]); UC1 --> UC2([System displays a list of old news articles]); UC2 --> UC3([Visitor selects a news article from the list]); UC3 --> UC4([System displays the selected old news article]); UC4 --> End((()))</pre>

HU-AC06	<p>@startuml start :Open News page; :Select "Subscribe to RSS"; :Copy or open RSS feed link in reader; stop @enduml</p>	 <pre>graph TD; Start(()) --> UC1([Open News page]); UC1 --> UC2([Select "Subscribe to RSS"]); UC2 --> UC3([Copy or open RSS feed link in reader]); UC3 --> End((()))</pre> <p>A UML Use Case Diagram for HU-AC06. It starts with a solid black circle (start node) pointing to a rounded rectangle labeled "Open News page". An arrow points down to another rounded rectangle labeled "Select 'Subscribe to RSS'". A third arrow points down to a rounded rectangle labeled "Copy or open RSS feed link in reader". Finally, an arrow points down to a bullseye symbol (end node).</p>	<p>@startuml start :Site member navigates to the news section; :Site member finds the RSS feed subscription option; :Site member clicks the subscribe button/link; :System provides the RSS feed URL; :Site member adds the feed URL to an RSS reader; stop @enduml</p>	 <pre>graph TD; Start(()) --> UC1([Site member navigates to the news section]); UC1 --> UC2([Site member finds the RSS feed subscription option]); UC2 --> UC3([Site member clicks the subscribe button/link]); UC3 --> UC4([System provides the RSS feed URL]); UC4 --> UC5([Site member adds the feed URL to an RSS reader]); UC5 --> End((()))</pre> <p>A UML Use Case Diagram for HU-AC06. It starts with a solid black circle (start node) pointing to a rounded rectangle labeled "Site member navigates to the news section". An arrow points down to another rounded rectangle labeled "Site member finds the RSS feed subscription option". A third arrow points down to a rounded rectangle labeled "Site member clicks the subscribe button/link". A fourth arrow points down to a rounded rectangle labeled "System provides the RSS feed URL". A fifth arrow points down to a rounded rectangle labeled "Site member adds the feed URL to an RSS reader". Finally, an arrow points down to a bullseye symbol (end node).</p>
---------	---	--	---	--

HU-AC07	<p>@startuml start :Open "Other Courses" page; :View list of upcoming courses; :Select a course of interest; :Read course details; stop @enduml</p>	 <pre>graph TD; Start(()) --> UC1[Open "Other Courses" page]; UC1 --> UC2[View list of upcoming courses]; UC2 --> UC3[Select a course of interest]; UC3 --> UC4[Read course details]; UC4 --> End((()))</pre> <p>A UML Use Case Diagram for HU-AC07. It starts with a solid black circle (start node) pointing to a rounded rectangle labeled "Open 'Other Courses' page". This is followed by a sequence of three rounded rectangles: "View list of upcoming courses", "Select a course of interest", and "Read course details". The final step points to a bullseye symbol (end node).</p>	<p>@startuml start :Site visitor navigates to the Other Courses section; :System displays a list of all upcoming Other Courses; :Visitor reviews the list of courses; :Visitor chooses a course; stop @enduml</p>	 <pre>graph TD; Start(()) --> UC1[Site visitor navigates to the Other Courses section]; UC1 --> UC2[System displays a list of all upcoming Other Courses]; UC2 --> UC3[Visitor reviews the list of courses]; UC3 --> UC4[Visitor chooses a course]; UC4 --> End((()))</pre> <p>A UML Use Case Diagram for HU-AC07. It starts with a solid black circle (start node) pointing to a rounded rectangle labeled "Site visitor navigates to the Other Courses section". This is followed by a sequence of three rounded rectangles: "System displays a list of all upcoming Other Courses", "Visitor reviews the list of courses", and "Visitor chooses a course". The final step points to a bullseye symbol (end node).</p>
---------	---	---	---	--

HU-AC08	<p>@startuml [Trainer] start :Open "Create course or event" page; :Enter course or event details; :Save and publish; [System] :Show confirmation; stop @enduml</p>	 <pre>graph TD subgraph Trainer Start(()) --> Open[Open "Create course or event" page] Open --> Enter[Enter course or event details] Enter --> Save[Save and publish] end subgraph System Save --> Show[Show confirmation] Show --> End(()) end</pre>	<p>@startuml start :Trainer initiates creation of a new course or event; if (Selects to create a 'Course'?) then (Yes) :Trainer enters course details; :Trainer saves the new course; else (No, creates 'Event') :Trainer enters event details; :Trainer saves the new event; endif :System makes the new course or event visible to site visitors; stop @enduml</p>	 <pre>graph TD Start(()) --> Init[Trainer initiates creation of a new course or event] Init --> Decision{Selects to create a 'Course'?} Decision -- Yes --> EnterCourse[Trainer enters course details] EnterCourse --> SaveCourse[Trainer saves the new course] Decision -- "No, creates 'Event'" --> EnterEvent[Trainer enters event details] EnterEvent --> SaveEvent[Trainer saves the new event] SaveCourse --> Merge{ } SaveEvent --> Merge Merge --> Visible[System makes the new course or event visible to site visitors] Visible --> End((()))</pre>
---------	---	--	--	--



HU-AC10	<p>@startuml [Trainer] start :Open "My courses or events"; :Select a course or event; :Delete item; :Confirm deletion; [System] :Remove from listings; :Show confirmation message; stop @enduml</p>	 <p>The diagram is a UML Use Case Diagram with two swimlanes: 'Trainer' and 'System'. The 'Trainer' swimlane contains four use cases: 'Open "My courses or events"', 'Select a course or event', 'Delete item', and 'Confirm deletion'. The 'System' swimlane contains two use cases: 'Remove from listings' and 'Show confirmation message'. Arrows indicate the flow: from 'Open "My courses or events"' to 'Select a course or event', then to 'Delete item', then to 'Confirm deletion'. An arrow from 'Confirm deletion' crosses the swimlane boundary to 'Remove from listings' in the 'System' swimlane, and then to 'Show confirmation message', which ends at a final state.</p>	<p>@startuml start :Trainer selects a course or event to delete; :Trainer requests deletion; :System asks for confirmation; if (Trainer confirms deletion?) then (Yes) :System deletes the course or event; :System removes the item from the list; :System displays a success message; else (No) :System cancels the deletion process; endif stop @enduml</p>	 <p>The diagram is a UML Activity Diagram. It starts with a start node leading to 'Trainer selects a course or event to delete'. This is followed by 'Trainer requests deletion' and 'System asks for confirmation'. A decision node 'Trainer confirms deletion?' branches the flow. The 'Yes' path leads to 'System deletes the course or event', then 'System removes the item from the list', and 'System displays a success message'. The 'No' path leads to 'System cancels the deletion process'. Both paths merge at a diamond node, which then leads to a final state.</p>
---------	--	---	--	---