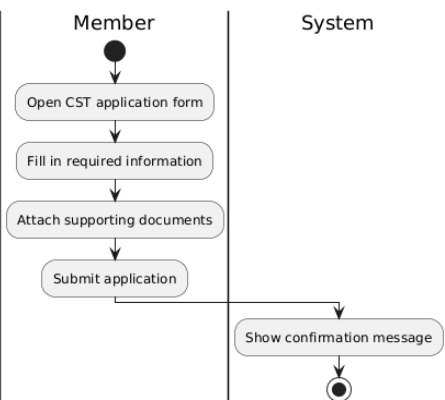
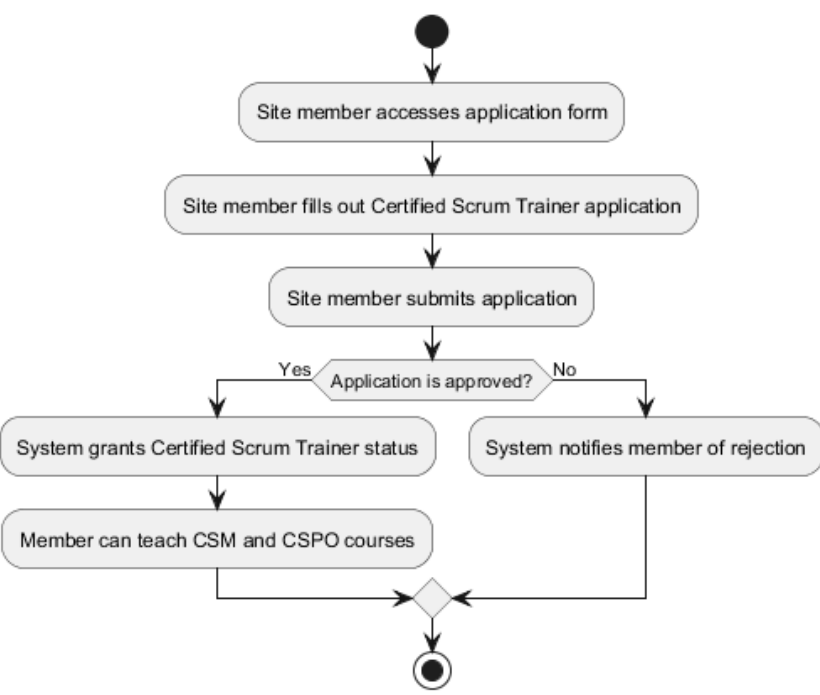
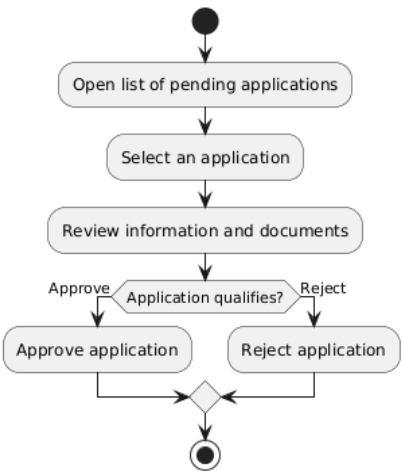
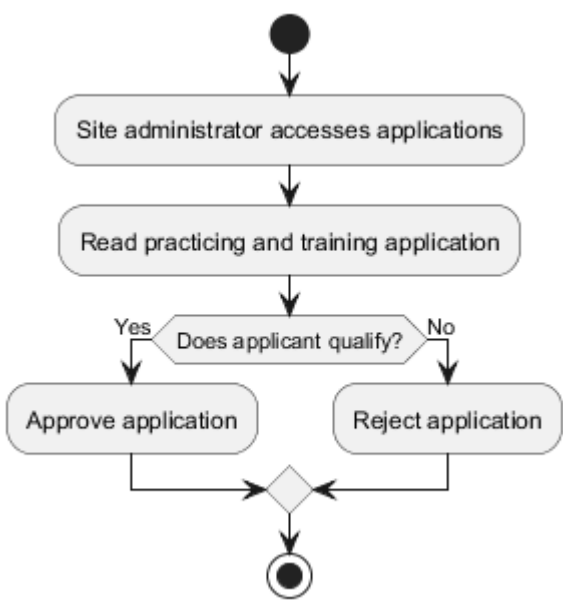
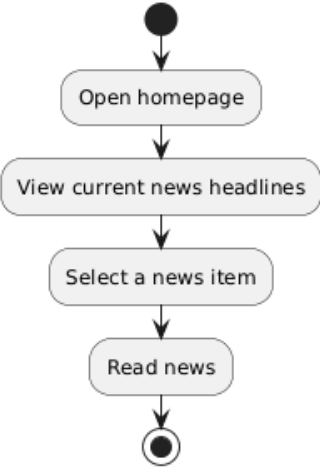
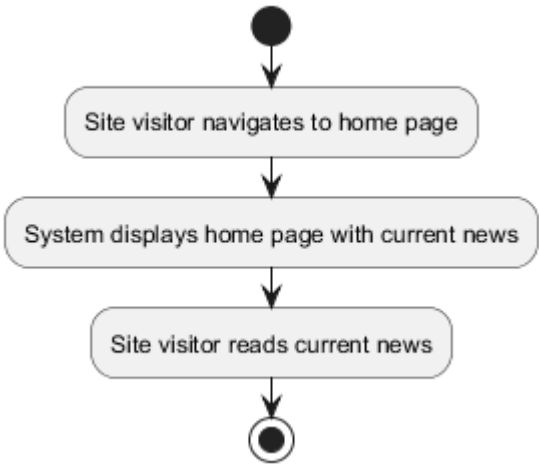
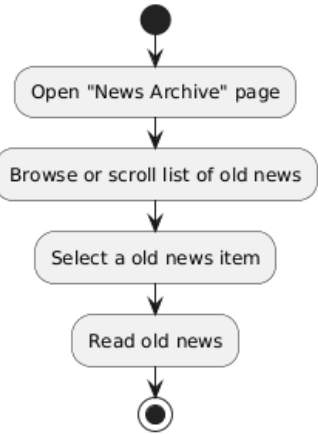
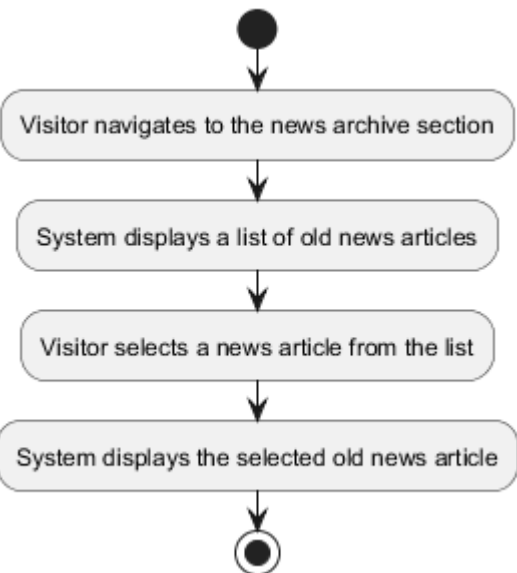
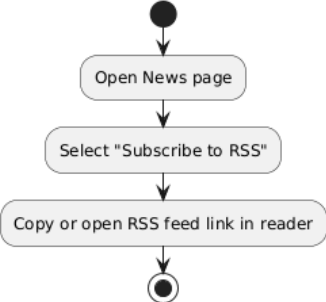
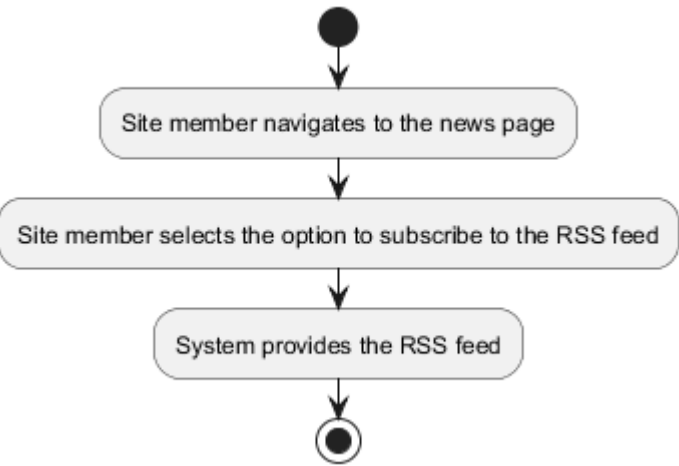


DIAGRAMAS DE REFERENCIA Y DIAGRAMAS GENERADOS

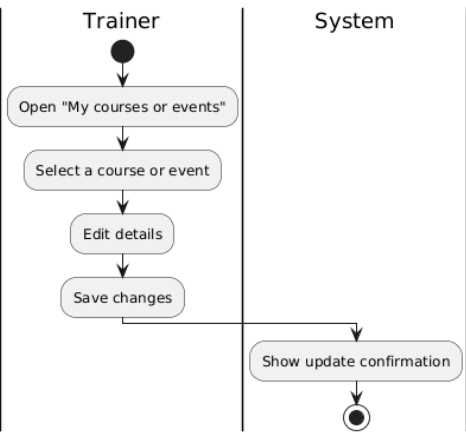
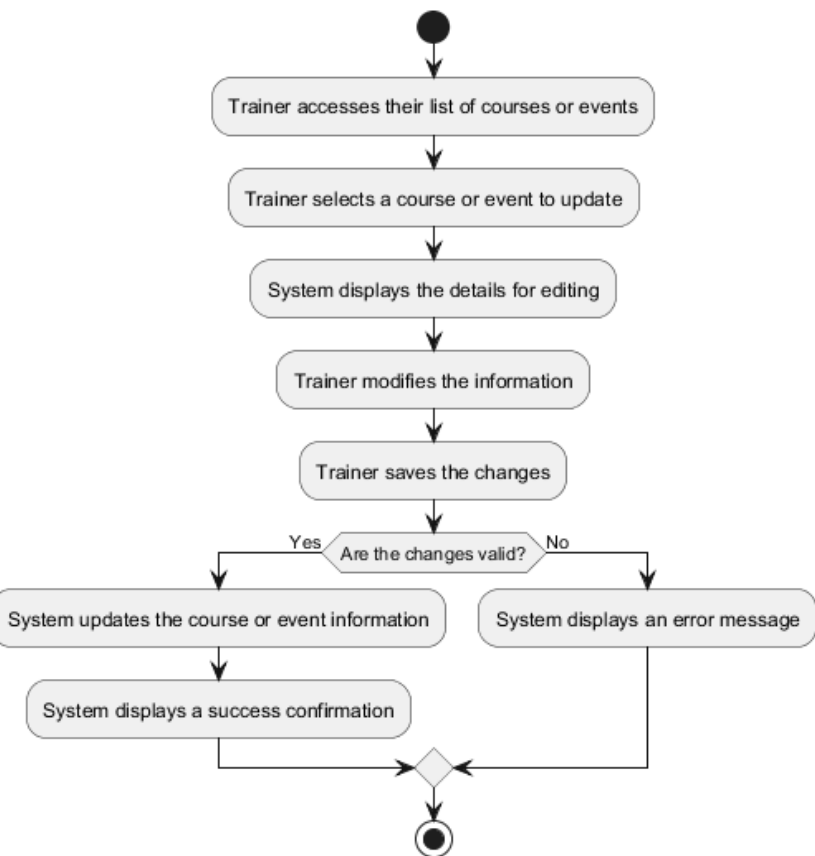
ID	Diagramas de Referencia		Diagramas Generados	
	Código en PlantUML	Diagrama en PlantUML	Código en PlantUML	Diagrama en PlantUML
HU-AC01	<pre>@startuml title Apply for Certified Scrum Practitioner (CSP)  [Member] start :Open "CSP Application";  [System] :Display application form;  [Member] :Fill required fields; :Attach evidence (files); :Preview application; :Submit;  [System] if (Input valid &amp; 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>		<pre>@startuml start :Access application for Certified Scrum Practitioner; :Fill out application; :Submit application; if (Application is approved?) then (Yes)   :Earn Certified Scrum Practitioner designation; else (No)   :Receive application status update; endif stop @enduml</pre>	

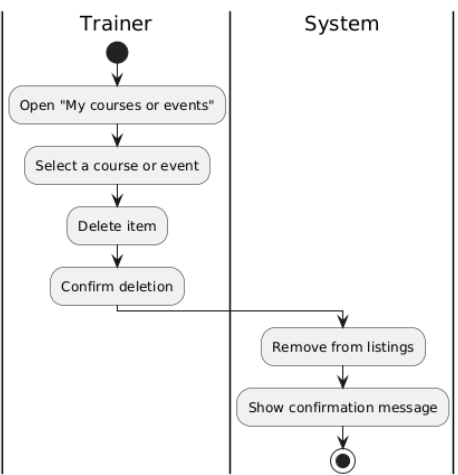
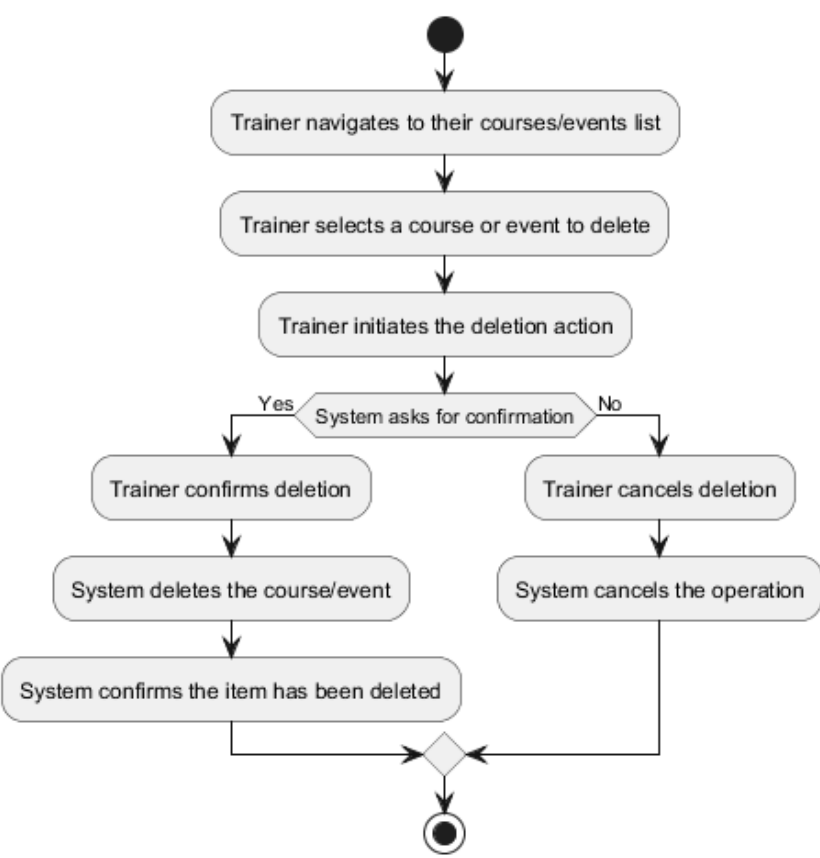
HU-AC02	<p>@startuml  Member  start :Open CST application form; :Fill in required information; :Attach supporting documents; :Submit application;   System  :Show confirmation message; stop @enduml</p>	 <pre>sequenceDiagram     participant Member     participant System     Member-&gt;&gt;Member: Open CST application form     Member-&gt;&gt;Member: Fill in required information     Member-&gt;&gt;Member: Attach supporting documents     Member-&gt;&gt;Member: Submit application     Member-&gt;&gt;System:      System-&gt;&gt;System: Show confirmation message     System--&gt;&gt;System: </pre>	<p>@startuml start :Site member accesses application form; :Site member fills out Certified Scrum Trainer application; :Site member submits application; if (Application is approved?) then (Yes)     :System grants Certified Scrum Trainer status;     :Member can teach CSM and CSPO courses; else (No)     :System notifies member of rejection; endif stop @enduml</p>	 <pre>graph TD     Start(( )) --&gt; A[Site member accesses application form]     A --&gt; B[Site member fills out Certified Scrum Trainer application]     B --&gt; C[Site member submits application]     C --&gt; D{Application is approved?}     D -- Yes --&gt; E[System grants Certified Scrum Trainer status]     D -- No --&gt; F[System notifies member of rejection]     E --&gt; G[Member can teach CSM and CSPO courses]     F --&gt; H{ }     G --&gt; H     H --&gt; 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(( )) --&gt; OpenList[Open list of pending applications]     OpenList --&gt; SelectApp[Select an application]     SelectApp --&gt; ReviewInfo[Review information and documents]     ReviewInfo --&gt; Qualifies{Application qualifies?}     Qualifies -- Approve --&gt; ApproveApp[Approve application]     Qualifies -- Reject --&gt; RejectApp[Reject application]     ApproveApp --&gt; Merge{ }     RejectApp --&gt; Merge     Merge --&gt; End((( )))</pre>	<p>@startuml start :Site administrator accesses applications; :Read practicing and training application; if (Does applicant qualify?) then (Yes)   :Approve application; else (No)   :Reject application; endif stop @enduml</p>	 <pre>graph TD     Start(( )) --&gt; AccessApps[Site administrator accesses applications]     AccessApps --&gt; ReadApp[Read practicing and training application]     ReadApp --&gt; Qualifies{Does applicant qualify?}     Qualifies -- Yes --&gt; ApproveApp[Approve application]     Qualifies -- No --&gt; RejectApp[Reject application]     ApproveApp --&gt; Merge{ }     RejectApp --&gt; Merge     Merge --&gt; 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(( )) --&gt; OpenHomepage[Open homepage]     OpenHomepage --&gt; ViewHeadlines[View current news headlines]     ViewHeadlines --&gt; SelectItem[Select a news item]     SelectItem --&gt; ReadNews[Read news]     ReadNews --&gt; End((( )))</pre>	<p>@startuml start :Site visitor navigates to home page; :System displays home page with current news; :Site visitor reads current news; stop @enduml</p>	 <pre>graph TD     Start(( )) --&gt; NavigateHome[Site visitor navigates to home page]     NavigateHome --&gt; DisplayNews[System displays home page with current news]     DisplayNews --&gt; ReadNews[Site visitor reads current news]     ReadNews --&gt; 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(( )) --&gt; UC1[Open "News Archive" page]; UC1 --&gt; UC2[Browse or scroll list of old news]; UC2 --&gt; UC3[Select a old news item]; UC3 --&gt; UC4[Read old news]; UC4 --&gt; End((( )))</pre>	<p>@startuml start :Visitor navigates to the news archive 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(( )) --&gt; UC1[Visitor navigates to the news archive section]; UC1 --&gt; UC2[System displays a list of old news articles]; UC2 --&gt; UC3[Visitor selects a news article from the list]; UC3 --&gt; UC4[System displays the selected old news article]; UC4 --&gt; 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(( )) --&gt; UC1[Open News page]; UC1 --&gt; UC2[Select "Subscribe to RSS"]; UC2 --&gt; UC3[Copy or open RSS feed link in reader]; UC3 --&gt; End((( )))</pre>	<p>@startuml start :Site member navigates to the news page; :Site member selects the option to subscribe to the RSS feed; :System provides the RSS feed; stop @enduml</p>	 <pre>graph TD; Start(( )) --&gt; UC1[Site member navigates to the news page]; UC1 --&gt; UC2[Site member selects the option to subscribe to the RSS feed]; UC2 --&gt; UC3[System provides the RSS feed]; UC3 --&gt; End((( )))</pre>

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(( )) --&gt; UC1[Open "Other Courses" page]; UC1 --&gt; UC2[View list of upcoming courses]; UC2 --&gt; UC3[Select a course of interest]; UC3 --&gt; UC4[Read course details]; UC4 --&gt; End((( )))</pre>	<p>@startuml start :Visitor requests to see upcoming Other Courses; :System retrieves the list of upcoming Other Courses; if (Are there any upcoming courses?) then (Yes) :System displays the list of all upcoming Other Courses; :Visitor views the list; :Visitor chooses the best course; else (No) :System displays a message indicating no upcoming courses; endif stop @enduml</p>	<pre>graph TD; Start(( )) --&gt; UC1[Visitor requests to see upcoming Other Courses]; UC1 --&gt; UC2[System retrieves the list of upcoming Other Courses]; UC2 --&gt; UC3{Are there any upcoming courses?}; UC3 -- Yes --&gt; UC4[System displays the list of all upcoming Other Courses]; UC3 -- No --&gt; UC5[System displays a message indicating no upcoming courses]; UC4 --&gt; UC6[Visitor views the list]; UC6 --&gt; UC7[Visitor chooses the best course]; UC7 --&gt; UC8{ }; UC5 --&gt; UC8; UC8 --&gt; End((( )))</pre>
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>sequenceDiagram; participant Trainer; participant System; Trainer-&gt;&gt;System: Open "Create course or event" page; Trainer-&gt;&gt;System: Enter course or event details; Trainer-&gt;&gt;System: Save and publish; System-&gt;&gt;Trainer: Show confirmation; Trainer-&gt;&gt;End: ;</pre>	<p>@startuml start :Trainer accesses the course creation interface; :Trainer enters details for the new course or event; :Trainer submits the new course or event; :System saves the new course or event; :System makes the course or event visible to site visitors; stop @enduml</p>	<pre>graph TD; Start(( )) --&gt; UC1[Trainer accesses the course creation interface]; UC1 --&gt; UC2[Trainer enters details for the new course or event]; UC2 --&gt; UC3[Trainer submits the new course or event]; UC3 --&gt; UC4[System saves the new course or event]; UC4 --&gt; UC5[System makes the course or event visible to site visitors]; UC5 --&gt; End((( )))</pre>

HU-AC09	<p>@startuml [Trainer] start :Open "My courses or events"; :Select a course or event; :Edit details; :Save changes;  [System] :Show update confirmation; stop @enduml</p>	 <p>The diagram shows two swimlanes: Trainer and System. The Trainer swimlane contains four use cases: 'Open "My courses or events"', 'Select a course or event', 'Edit details', and 'Save changes', connected sequentially by downward arrows. The System swimlane contains one use case: 'Show update confirmation'. An arrow points from the 'Save changes' use case in the Trainer swimlane to the 'Show update confirmation' use case in the System swimlane. The diagram ends with a final bullseye symbol in the System swimlane.</p>	<p>@startuml start :Trainer accesses their list of courses or events; :Trainer selects a course or event to update; :System displays the details for editing; :Trainer modifies the information; :Trainer saves the changes; if (Are the changes valid?) then (Yes)   :System updates the course or event information;   :System displays a success confirmation; else (No)   :System displays an error message; endif stop @enduml</p>	 <p>The diagram is a single swimlane activity diagram. It starts with a start node (solid black circle) leading to 'Trainer accesses their list of courses or events'. This is followed by 'Trainer selects a course or event to update', then 'System displays the details for editing', then 'Trainer modifies the information', and finally 'Trainer saves the changes'. A decision diamond labeled 'Are the changes valid?' follows. The 'Yes' path leads to 'System updates the course or event information', which then leads to 'System displays a success confirmation'. The 'No' path leads to 'System displays an error message'. Both paths merge at a diamond, which then leads to a final bullseye symbol.</p>
---------	---	---	---	--

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 the start node to 'Open "My courses or events"', then to 'Select a course or event', then to 'Delete item', then to 'Confirm deletion'. An arrow connects 'Confirm deletion' to 'Remove from listings'. Another arrow connects 'Remove from listings' to 'Show confirmation message', which ends at a final node.</p>	<p>@startuml start :Trainer navigates to their courses/events list; :Trainer selects a course or event to delete; :Trainer initiates the deletion action; if (System asks for confirmation) then (Yes) :Trainer confirms deletion; :System deletes the course/event; :System confirms the item has been deleted; else (No) :Trainer cancels deletion; :System cancels the operation; endif stop @enduml</p>	 <p>The diagram is a UML Activity Diagram. It starts with a start node leading to 'Trainer navigates to their courses/events list'. This is followed by 'Trainer selects a course or event to delete', then 'Trainer initiates the deletion action'. A decision node 'System asks for confirmation' follows. The 'Yes' path leads to 'Trainer confirms deletion', then 'System deletes the course/event', and finally 'System confirms the item has been deleted'. The 'No' path leads to 'Trainer cancels deletion', then 'System cancels the operation'. Both paths merge at a join node (diamond) before reaching the final node.</p>
---------	--	--	---	---