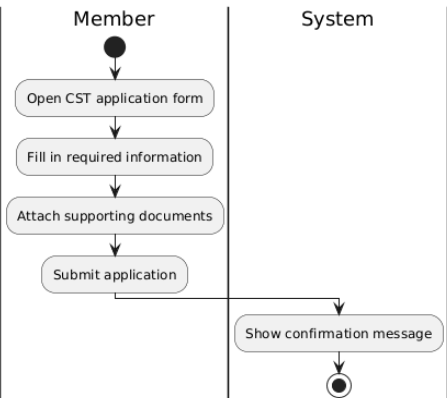
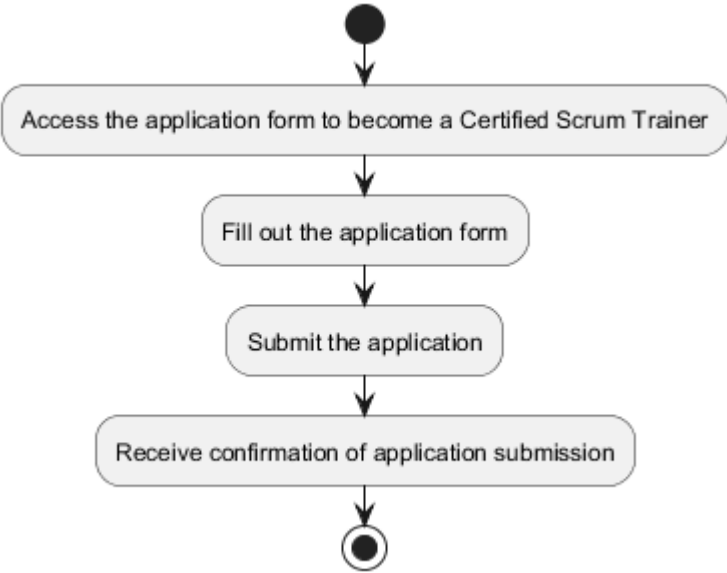
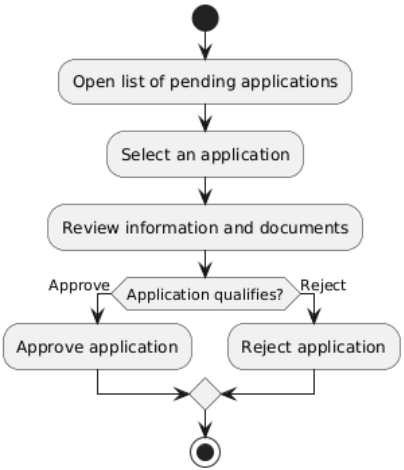
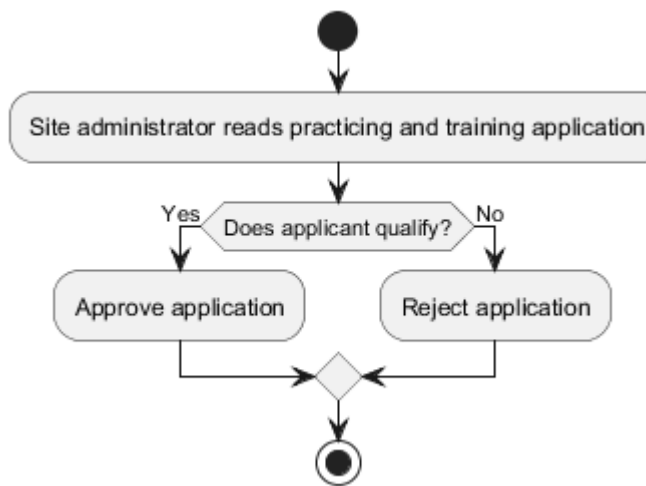
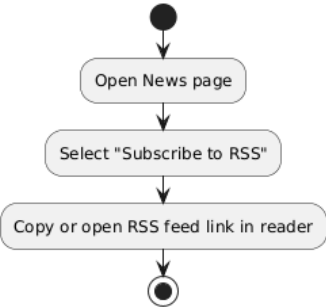
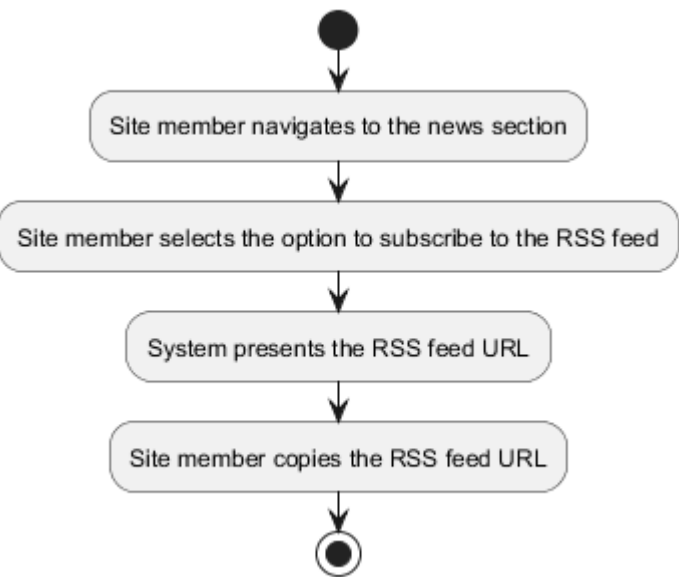


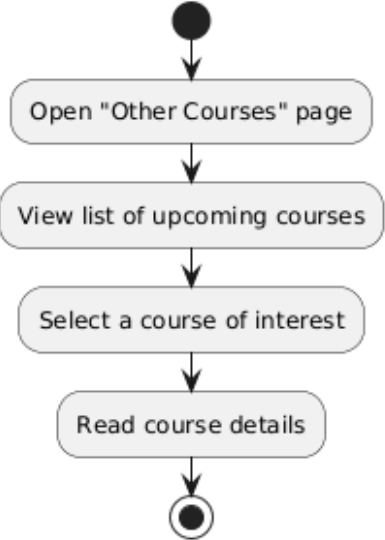
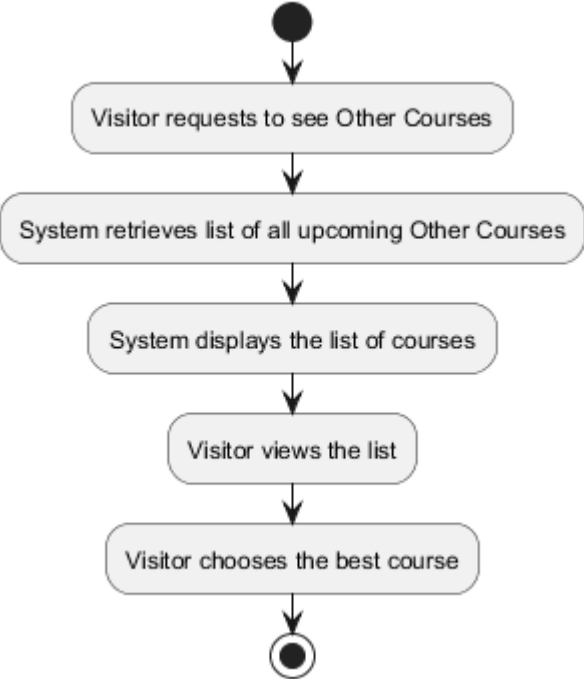
DIAGRAMAS DE REFERENCIA Y DIAGRAMAS GENERADOS

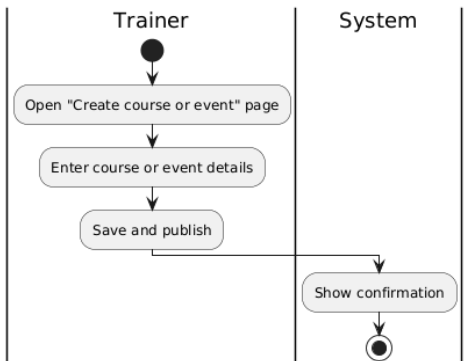
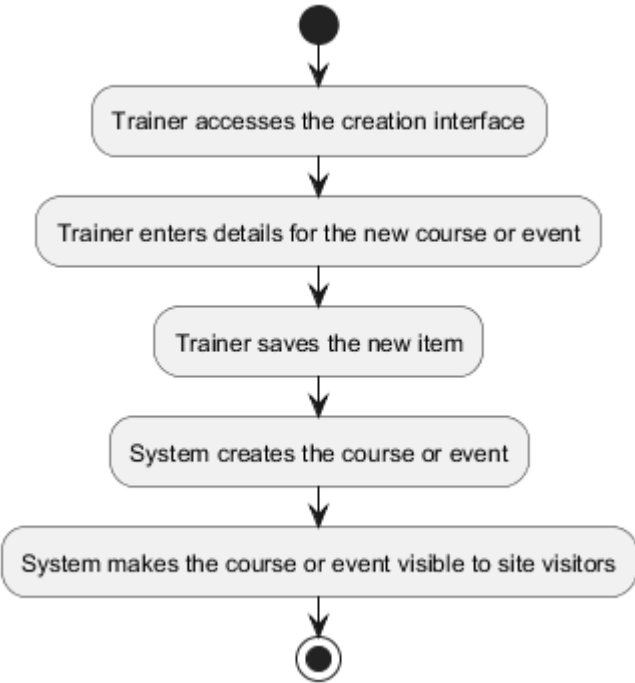
ID	Diagramas de Referencia (Humano)		Diagramas Generados (Herramienta)	
	Código en PlantUML	Diagrama en PlantUML	Código en PlantUML	Diagrama en PlantUML
HU-AC01	<pre>@startuml title Apply for Certified Scrum Practitioner (CSP) [Member] start :Open "CSP Application"; [System] :Display application form; [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 with 'Persist application', 'Assign trackingId', 'Queue confirmation email', and 'Show success message' before ending. If 'No', it performs 'Show inline errors' and 'Keep entered data' before returning to the Member to start the process again.</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 an initial 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 final node.</p>

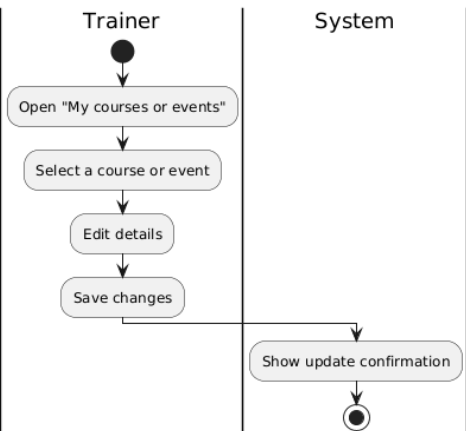

HU-AC02	<p>@startuml Member start :Open CST application form; :Fill in required information; :Attach supporting documents; :Submit application; System :Show confirmation message; stop @enduml</p>	 <pre>sequenceDiagram participant Member participant System Member->>Open CST application form Member->>Fill in required information Member->>Attach supporting documents Member->>Submit application System->>Show confirmation message</pre>	<p>@startuml start :Access the application form to become a Certified Scrum Trainer; :Fill out the application form; :Submit the application; :Receive confirmation of application submission; stop @enduml</p>	 <pre>graph TD Start(()) --> UC1[Access the application form to become a Certified Scrum Trainer] UC1 --> UC2[Fill out the application form] UC2 --> UC3[Submit the application] UC3 --> UC4[Receive confirmation of application submission] UC4 --> 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(()) --> UC1[Open list of pending applications] UC1 --> UC2[Select an application] UC2 --> UC3[Review information and documents] UC3 --> UC4{Application qualifies?} UC4 -- Approve --> UC5[Approve application] UC4 -- Reject --> UC6[Reject application] UC5 --> End((())) UC6 --> End</pre>	<p>@startuml start :Site administrator reads practicing and training application; if (Does applicant qualify?) then (Yes) :Approve application; else (No) :Reject application; endif stop @enduml</p>	 <pre>graph TD Start(()) --> UC1[Site administrator reads practicing and training application] UC1 --> UC2{Does applicant qualify?} UC2 -- Yes --> UC3[Approve application] UC2 -- No --> UC4[Reject application] UC3 --> End((())) UC4 --> 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>A UML Use Case Diagram for HU-AC04. It starts with a solid black circle, followed by a sequence of four rounded rectangular use cases: 'Open homepage', 'View current news headlines', 'Select a news item', and 'Read news'. The sequence ends with a bullseye symbol.</p>	<p>@startuml start :Visitor navigates to the home page; :System displays current news; :Visitor reads the news; stop @enduml</p>	<pre>graph TD; Start(()) --> UC1([Visitor navigates to the home page]); UC1 --> UC2([System displays current news]); UC2 --> UC3([Visitor reads the news]); UC3 --> End((()))</pre> <p>A UML Use Case Diagram for HU-AC04. It starts with a solid black circle, followed by a sequence of three rounded rectangular use cases: 'Visitor navigates to the home page', 'System displays current news', and 'Visitor reads the news'. The sequence ends with a bullseye symbol.</p>
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>A UML Use Case Diagram for HU-AC05. It starts with a solid black circle, followed by a sequence of four rounded rectangular use cases: 'Open "News Archive" page', 'Browse or scroll list of old news', 'Select a old news item', and 'Read old news'. The sequence ends with a bullseye symbol.</p>	<p>@startuml start :Visitor navigates to the news archive section; :Visitor searches or browses for an old news article; if (Desired article is found?) then (Yes) :Visitor selects the news article; :System displays the selected old news article; else (No) :System indicates that the article was not found; endif stop @enduml</p>	<pre>graph TD; Start(()) --> UC1([Visitor navigates to the news archive section]); UC1 --> UC2([Visitor searches or browses for an old news article]); UC2 --> Decision{Desired article is found?}; Decision -- Yes --> UC3([Visitor selects the news article]); UC3 --> UC4([System displays the selected old news article]); Decision -- No --> UC5([System indicates that the article was not found]); UC4 --> Merge(()); UC5 --> Merge; Merge --> End((()))</pre> <p>A UML Use Case Diagram for HU-AC05. It starts with a solid black circle, followed by two rounded rectangular use cases: 'Visitor navigates to the news archive section' and 'Visitor searches or browses for an old news article'. This is followed by a decision diamond labeled 'Desired article is found?'. If 'Yes', the flow goes to 'Visitor selects the news article' and then 'System displays the selected old news article'. If 'No', the flow goes to 'System indicates that the article was not found'. Both paths merge at a diamond before the final bullseye end node.</p>

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 down 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 selects the option to subscribe to the RSS feed; :System presents the RSS feed URL; :Site member copies the RSS feed URL; stop @enduml</p>	 <pre>graph TD; Start(()) --> UC1([Site member navigates to the news section]); UC1 --> UC2([Site member selects the option to subscribe to the RSS feed]); UC2 --> UC3([System presents the RSS feed URL]); UC3 --> UC4([Site member copies the RSS feed URL]); UC4 --> End((()))</pre> <p>A UML Use Case Diagram for HU-AC06. It starts with a solid black circle (start node) pointing down to a rounded rectangle labeled "Site member navigates to the news section". An arrow points down to another rounded rectangle labeled "Site member selects the option to subscribe to the RSS feed". A third arrow points down to a rounded rectangle labeled "System presents the RSS feed URL". A fourth arrow points down to a rounded rectangle labeled "Site member copies the RSS feed URL". 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(()) --> A[Open "Other Courses" page]; A --> B[View list of upcoming courses]; B --> C[Select a course of interest]; C --> D[Read course details]; D --> End((()))</pre> <p>The diagram is a vertical flowchart starting with a solid black circle. It contains four rounded rectangular boxes: "Open 'Other Courses' page", "View list of upcoming courses", "Select a course of interest", and "Read course details". These boxes are connected by downward-pointing arrows. The flow ends at a bullseye symbol.</p>	<p>@startuml start :Visitor requests to see Other Courses; :System retrieves list of all upcoming Other Courses; :System displays the list of courses; :Visitor views the list; :Visitor chooses the best course; stop @enduml</p>	 <pre>graph TD; Start(()) --> A[Visitor requests to see Other Courses]; A --> B[System retrieves list of all upcoming Other Courses]; B --> C[System displays the list of courses]; C --> D[Visitor views the list]; D --> E[Visitor chooses the best course]; E --> End((()))</pre> <p>The diagram is a vertical flowchart starting with a solid black circle. It contains five rounded rectangular boxes: "Visitor requests to see Other Courses", "System retrieves list of all upcoming Other Courses", "System displays the list of courses", "Visitor views the list", and "Visitor chooses the best course". These boxes are connected by downward-pointing arrows. The flow ends at a bullseye symbol.</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 direction TB T1(()) --> T1_1[Open "Create course or event" page] T1_1 --> T1_2[Enter course or event details] T1_2 --> T1_3[Save and publish] end subgraph System direction TB S1[Show confirmation] --> S1_1((())) end T1_3 --> S1</pre>	<p>@startuml start :Trainer accesses the creation interface; :Trainer enters details for the new course or event; :Trainer saves the new item; :System creates the course or event; :System makes the course or event visible to site visitors; stop @enduml</p>	 <pre>graph TD Start(()) --> A[Trainer accesses the creation interface] A --> B[Trainer enters details for the new course or event] B --> C[Trainer saves the new item] C --> D[System creates the course or event] D --> E[System makes the course or event visible to site visitors] E --> 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>	 <pre>graph TD subgraph Trainer Start(()) --> Open[Open "My courses or events"] Open --> Select[Select a course or event] Select --> Edit[Edit details] Edit --> Save[Save changes] end subgraph System Save --> Confirm[Show update confirmation] Confirm --> End((())) end</pre>	<p>@startuml start :Trainer navigates to their courses list; :Trainer selects a course or event to update; :System displays the course or event details for editing; :Trainer modifies the information; :Trainer saves the changes; :System updates the course or event information; :System displays a confirmation message; stop @enduml</p>	 <pre>graph TD Start(()) --> A[Trainer navigates to their courses list] A --> B[Trainer selects a course or event to update] B --> C[System displays the course or event details for editing] C --> D[Trainer modifies the information] D --> E[Trainer saves the changes] E --> F[System updates the course or event information] F --> G[System displays a confirmation message] G --> End((()))</pre>
---------	---	--	--	--

