

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>Apply for Certified Scrum Practitioner (CSP)</p> <pre> graph TD Member((Member)) --> OpenCSP[Open "CSP Application"] OpenCSP --> DisplayForm[Display application form] System((System)) --> DisplayForm DisplayForm --> FillFields[Fill required fields] FillFields --> AttachEvidence[Attach evidence (files)] AttachEvidence --> Preview[Preview application] Preview --> Submit[Submit] Submit --> Decision{Input valid & files ok?} Decision -- Yes --> Persist[Persist application] Persist --> Assign[Assign trackingId] Assign --> Queue[Queue confirmation email] Queue --> Success[Show success message] Success --> End(()) Decision -- No --> Errors[Show inline errors] Errors --> KeepData[Keep entered data] KeepData --> End </pre>	<pre> @startuml start :Access the application form for Certified Scrum Practitioner; :Fill out the application; :Submit the application; stop @enduml </pre>	<pre> graph TD Start(()) --> Access[Access the application form for Certified Scrum Practitioner] Access --> Fill[Fill out the application] Fill --> Submit[Submit the application] Submit --> End(()) </pre>

HU-AC02	<pre> @startuml Member start :Open CST application form; :Fill in required information; :Attach supporting documents; :Submit application; System :Show confirmation message; stop @enduml </pre>	<pre> sequenceDiagram participant Member participant System Member->>System: Open CST application form activate System Member->>System: Fill in required information deactivate System Member->>System: Attach supporting documents deactivate System Member->>System: Submit application deactivate System System-->>Member: Show confirmation message </pre>	<pre> @startuml start :Site member accesses application form; :Site member fills out application; :Site member submits application; if (Application is approved?) then (Yes) :Become a Certified Scrum Trainer; else (No) :Receive application rejection; endif stop @enduml </pre>	<pre> graph TD Start(()) --> Access[Site member accesses application form] Access --> Fill[Site member fills out application] Fill --> Submit[Site member submits application] Submit --> Decision{Application is approved?} Decision -- Yes --> Become[Become a Certified Scrum Trainer] Decision -- No --> Reject[Receive application rejection] Become --> End((())) Reject --> End </pre>
HU-AC03	<pre> @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 </pre>	<pre> sequenceDiagram participant Admin Admin->>Admin: Open list of pending applications Admin->>Admin: Select an application Admin->>Admin: Review information and documents Admin-->>Decision{Application qualifies?} Decision -- Approve --> Admin: Approve application Decision -- Reject --> Admin: Reject application </pre>	<pre> @startuml start :Administrator reads practicing and training application; if (Does the applicant qualify?) then (Yes) :Administrator approves the application; else (No) :Administrator rejects the application; endif stop @enduml </pre>	<pre> graph TD Start(()) --> Read[Administrator reads practicing and training application] Read --> Decision{Does the applicant qualify?} Decision -- Yes --> Approve[Administrator approves the application] Decision -- No --> Reject[Administrator rejects the application] Approve --> End((())) Reject --> End </pre>

HU-AC04	<pre>@startuml start :Open homepage; :View current news headlines; :Select a news item; :Read news; stop @enduml</pre>	<pre> graph TD Start(()) --> OpenHomepage[Open homepage] OpenHomepage --> ViewHeadlines[View current news headlines] ViewHeadlines --> SelectItem[Select a news item] SelectItem --> ReadNews[Read news] ReadNews --> End((())) </pre>	<pre>@startuml start :Site visitor navigates to the home page; :System displays the home page with current news; :Site visitor reads current news; stop @enduml</pre>	<pre> graph TD Start(()) --> SiteVisitorNavigates[Site visitor navigates to the home page] SiteVisitorNavigates --> SystemDisplays[System displays the home page with current news] SystemDisplays --> SiteVisitorReads[Site visitor reads current news] SiteVisitorReads --> End((())) </pre>
HU-AC05	<pre>@startuml start :Open "News Archive" page; :Browse or scroll list of old news; :Select a old news item; :Read old news; stop @enduml</pre>	<pre> graph TD Start(()) --> OpenArchive[Open "News Archive" page] OpenArchive --> BrowseOldNews[Browse or scroll list of old news] BrowseOldNews --> SelectOldItem[Select a old news item] SelectOldItem --> ReadOldNews[Read old news] ReadOldNews --> End((())) </pre>	<pre>@startuml start :Visitor navigates to the news archive page; :Visitor searches or browses for old news; if (Is the desired news item found?) then (Yes) :System displays the selected old news item; :Visitor reads the news content; else (No) :System displays a 'not found' message; endif stop @enduml</pre>	<pre> graph TD Start(()) --> VisitorNavigates[Visitor navigates to the news archive page] VisitorNavigates --> VisitorSearches[Visitor searches or browses for old news] VisitorSearches --> Decision{Is the desired news item found?} Decision -- Yes --> SystemDisplays[System displays the selected old news item] SystemDisplays --> VisitorReads[Visitor reads the news content] Decision -- No --> SystemDisplaysNotFound[System displays a 'not found' message] VisitorReads --> End((())) SystemDisplaysNotFound --> End((())) </pre>

HU-AC06	<pre>@startuml start :Open News page; :Select "Subscribe to RSS"; :Copy or open RSS feed link in reader; stop @enduml</pre>	<pre> graph TD Start(()) --> Open[Open News page] Open --> Select[Select "Subscribe to RSS"] Select --> Copy[Copy or open RSS feed link in reader] Copy --> End((())) </pre>	<pre>@startuml start :Site member navigates to news section; :Site member selects option to subscribe to RSS feed; :System provides the RSS feed link; :Site member adds the feed link to their RSS reader; stop @enduml</pre>	<pre> graph TD Start(()) --> SiteMember[Site member navigates to news section] SiteMember --> Select[Site member selects option to subscribe to RSS feed] Select --> System[System provides the RSS feed link] System --> Member[Site member adds the feed link to their RSS reader] Member --> End((())) </pre>
HU-AC07	<pre>@startuml start :Open "Other Courses" page; :View list of upcoming courses; :Select a course of interest; :Read course details; stop @enduml</pre>	<pre> graph TD Start(()) --> Open[Open "Other Courses" page] Open --> View[View list of upcoming courses] View --> Select[Select a course of interest] Select --> Read[Read course details] Read --> End((())) </pre>	<pre>@startuml start :Visitor navigates to the Other Courses section; :System retrieves the list of upcoming Other Courses; if (Are there any upcoming courses?) then (Yes) :System displays the list of upcoming Other Courses; :Visitor reviews the list; :Visitor chooses a course; else (No) :System displays a message indicating no upcoming courses; endif stop @enduml</pre>	<pre> graph TD Start(()) --> Visitor[Visitor navigates to the Other Courses section] Visitor --> Retrieve[System retrieves the list of upcoming Other Courses] Retrieve --> Decision{Are there any upcoming courses?} Decision -- Yes --> DisplayList[System displays the list of upcoming Other Courses] Decision -- No --> DisplayMessage[System displays a message indicating no upcoming courses] DisplayList --> Review[Visitor reviews the list] Review --> Choose[Visitor chooses a course] Choose --> End((())) </pre>

HU-AC08	<pre> @startuml Trainer start :Open "Create course or event" page; :Enter course or event details; :Save and publish; System :Show confirmation; stop @enduml </pre>	<pre> sequenceDiagram participant Trainer participant System Trainer->>System: Open "Create course or event" page Trainer->>System: Enter course or event details Trainer->>System: Save and publish System->>Trainer: Show confirmation stop </pre>	<pre> @startuml start :Trainer accesses the course creation form; :Trainer fills in the new course or event details; :Trainer submits the form; :System creates the new course or event; :System makes the course or event visible to site visitors; :System confirms the creation; stop @enduml </pre>	<pre> graph TD Start(()) --> Access[Trainer accesses the course creation form] Access --> Fill[Trainer fills in the new course or event details] Fill --> Submit[Trainer submits the form] Submit --> Create[System creates the new course or event] Create --> Visible[System makes the course or event visible to site visitors] Visible --> Confirmation[System confirms the creation] Confirmation --> End((())) </pre>
---------	---	--	---	--

HU-AC09	<pre> @startuml Trainer start :Open "My courses or events"; :Select a course or event; >Edit details; :Save changes; System :Show update confirmation; stop @enduml </pre>	<pre> sequenceDiagram participant Trainer participant System Trainer->>System: Open "My courses or events" activate System Trainer->>System: Select a course or event activate System Trainer->>System: Edit details activate System Trainer->>System: Save changes deactivate System System->>Trainer: Show update confirmation deactivate Trainer </pre>	<pre> @startuml start :Trainer navigates to their courses or events list; :Trainer selects a course or event to update; :System displays the editable details; :Trainer modifies the information; :Trainer saves the changes; if (Information is valid?) then (Yes) :System updates the course or event; :System displays a success message; else (No) :System displays an error message; endif stop @enduml </pre>	<pre> activityDiagram [*] --> Trainer [*] --> System Trainer --> System: Trainer navigates to their courses or events list Trainer --> System: Trainer selects a course or event to update System --> Trainer: System displays the editable details Trainer --> System: Trainer modifies the information Trainer --> System: Trainer saves the changes [*] --> Decision: Information is valid? Decision --> IfYes: System updates the course or event Decision --> IfYes: System --> Trainer: System displays a success message Decision --> IfNo: System --> Trainer: System displays an error message IfYes --> Decision IfNo --> Decision Trainer --> [*]: loop back to start </pre>
---------	--	--	---	---

HU-AC10	<pre> @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 </pre>	<pre> sequenceDiagram participant Trainer participant System Trainer->>System: Open "My courses or events" activate System System->>Trainer: Select a course or event Trainer->>System: Delete item activate System System->>Trainer: Confirm deletion deactivate System Trainer->>System: Remove from listings deactivate System System-->>Trainer: Show confirmation message deactivate Trainer </pre>	<pre> @startuml start :Trainer navigates to their list of courses or events; :Trainer selects a course or event to delete; :Trainer requests to delete the selected item; if (Trainer confirms deletion?) then (Yes) :System deletes the course or event; :System confirms the item has been deleted; :System removes the item from the list; else (No) :System cancels the deletion request; endif stop @enduml </pre>	<pre> graph TD Start(()) --> Nav[Trainer navigates to their list of courses or events] Nav --> Sel[Trainer selects a course or event to delete] Sel --> Req[Trainer requests to delete the selected item] Req --> Cond{Trainer confirms deletion?} Cond -- Yes --> Del[System deletes the course or event] Del --> Conf[System confirms the item has been deleted] Conf --> Rem[System removes the item from the list] Rem --> End((())) Cond -- No --> Cancel[System cancels the deletion request] End --> Rem Rem --> Decision(()) Decision --> End </pre>
---------	--	--	---	---