|  |  |
| --- | --- |
| CLIENT | GreenSQA |
| USER | GreenSQA company collaborators and project managers |
| functional requirements | * Create a project. * Culminate a stage of a project. * Register capsule. * Approve capsule. * Publish capsule. * Report of registered capsules by type. * Report of lessons learned corresponding to the capsules registered in a stage. * Project report with more registered capsules. * Report of capsules registered by a collaborator. * Search for situations and capsule lessons approved and published in a search string found in the hashtags. * Obtain the hashtags of the lessons and situations. |
| Problem Context | GreenSQA requests a system in which information can be retained  of employees before they rotate to other employees. for this his  main route will be the capsules of knowledge, in which the  employees will be able to fill the capsules with pertinent information  (situations, lessons and important data) to their projects. The system  It will use two users, the managers (Focused on the management  of projects, approval of capsules and the generation of  reports) and collaborators (Focused on the creation and consultation of  capsules), where each one will have its own methods (requirements) and  will be shared by others, the program should be useful for managing projects and  its stages, record the knowledge capsules, approve them,  publish, consult and report them. |
| non-functional requirements | * Scalability: The system must be able to handle a large number of knowledge capsules, since many of them are expected to be generated over time.      * Performance: The system must be able to handle multiple user requests at the same time, without long wait times.      * Usability: The system must be easy to use and intuitive for end users, with a clear and well-designed interface that allows easy navigation through the different functionalities.      * Flexibility: The system must allow the customization of certain aspects, such as the configuration of the types of knowledge capsules or the definition of the keywords that will be used for the hashtag.      * Integration: The system must be compatible with other company tools to guarantee greater efficiency and reduce the duplication of information.      * Maintenance: The system must be easy to maintain and update, with clear and complete documentation for the personnel in charge of its administration and maintenance.      * Traceability: The system must allow the traceability of the knowledge capsules, from their creation to their approval and publication, to guarantee greater transparency in the process and better knowledge management. |

| Name or identifier | FR1: Create a project | | |
| --- | --- | --- | --- |
| Summary | The manager is asked to enter the corresponding data to create the project. The project is created within the system and the creation confirmation is printed. | | |
| Inputs | Input name | Datatype | Selection condition |
| nameProject | String |  |
| startDateProject | Calendar | yyyy-MM-dd |
| endDateProject | Calendar | yyyy-MM-dd |
| budget | double | $ |
| greenSQAManagerName | String |  |
|  | greenSQAManagerPhone | String | ###-###-#### |
|  | companyManagerName | String |  |
|  | companyManagerPhone | String | ###-###-#### |
|  | monthsStage | int[] |  |
| Result | The project is created with the name of the project, its start/end dates, budget, its managers are created, the duration in months of each stage is stored, creating its 6 stages. Then, the message of the created project number is printed. | | |
| Outputs | output name | Datatype | Selection condition |
| creationProjectConfirmation | String | “The project ” +number of project+ “created” |
|
|
|

| Name or identifier | FR2: Culminate a stage of a project | | |
| --- | --- | --- | --- |
| Summary | The manager is prompted to enter the project number, plus the day the current stage ended to start the next stage. The system edits the previous stage leaving it inactive and the new stage remains activated until the next completion. Then, the date on which the new stage must be completed is printed to comply with the number of months for each stage. | | |
| Inputs | Input name | Datatype | Selection condition |
| searchProject | int | 0-9 |
| startNewStageReal | Calendar | yyyy-MM-dd |
| Result | The project is obtained, the current stage is inactive and the immediately following one is activated. The start date of the new stage is saved and the date when the stage should end is printed. | | |
| Outputs | output name | Datatype | Selection condition |
| futureEndStage | Calendar | dd-MM-yyyy |
|
|
|

| Name or identifier | FR3: Register Capsule | | |
| --- | --- | --- | --- |
| Summary | The collaborator enters the necessary data to create the capsule, this is entered into the selected project and the active stage. Then, a registration confirmation message is printed. | | |
| Inputs | Input name | Datatype | Selection condition |
| situation | String | Have to contain 1-3 keywords between ##, Example: Hey #Hello world# How are #you# ? |
| typeCapsule | int | 0-3 |
| authorCapsule | String |  |
| positionAuthor | String |  |
| lessonCapsule | String | have to contain 1-3 keywords between ##, Example: Hey #Hello world# How are #you# ? |
| searchProject | int | 0-9 |
| Result | The situation, type, author, position of the author and lesson of the capsule in the selected project and the active stage are recorded. A confirmation message is then printed. | | |
| Outputs | output name | Datatype | Selection condition |
| creationCapsuleConfirmation | String | “The capsule +(numberCapsule)+ for the project + (numberProject)+ in the stage +(numberStage)+ has been created. |
|
|
|

| Name or identifier | FR4: Approve capsule | | |
| --- | --- | --- | --- |
| Summary | The manager must enter the data to find the capsule to approve. The system approves the capsule and the approval date is saved. | | |
| Inputs | Input name | Datatype | Selection condition |
| searchProject | int | 0-9 |
| calendarTime | Calendar | yyyy-MM-dd |
| stage | int | 0-5 |
| capsule | int | 0-49 |
|  |  |  |
| Result | The capsule is searched and if it has the data entered, it is approved to be published. Then, the date on which the capsule was approved is printed. | | |
| Outputs | output name | Datatype | Selection condition |
| timeStamp | Calendar | dd-MM-yyyy |
|
|
|

| Name or identifier | FR5: Publish capsule | | |
| --- | --- | --- | --- |
| Summary | The manager enters the location of the capsule, the system confirms that the capsule has been approved, if it is approved, the url where the capsule is located will be printed in HTML format. | | |
| Inputs | Input name | Datatype | Selection condition |
| searchProject | int | 0-9 |
| stage | int | 0-5 |
| capsule | int | 0-49 |
|  |  |  |
|  |  |  |
| Result | The system looks for the capsule with the location of the project, stage and capsule to verify that it has already been approved. Then, it is published by adding the url where the published capsule is located. | | |
| Outputs | output name | Datatype | Selection condition |
| URLpost | String | “https://github.com/capsules/xxxxxxx” |
|
|
|

| Name or identifier | FR6: Report of registered capsules by type. | | |
| --- | --- | --- | --- |
| Summary | The System asks the user to enter the Project to search. The system makes a loop to obtain the number of capsules of each type, printing the number of each type at the end of the loop. | | |
| Inputs | Input name | Datatype | Selection condition |
| project | int | 0-9 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Result | The program obtains the amount of each type of capsule in the Project entered and prints a message with the results found. | | |
| Outputs | output name | Datatype | Selection condition |
| amountCapsules | String | "The amount of tecnic capsules is: " + tecnicAmount + "The amount of management capsules is: " + managementAmount+ "The amount of domain capsules is: " + domainAmount +"The amount of experience capsules is: " + experienceAmount |
|
|
|

| Name or identifier | FR7: Report of lessons learned corresponding to the capsules registered in a stage | | |
| --- | --- | --- | --- |
| Summary | The System asks the user to enter a project and stage to search, the program finds all the lessons within the stage and prints them until the existing ones are finished. | | |
| Inputs | Input name | Datatype | Selection condition |
| project | int | 0-9 |
| stage | int | 0-5 |
|  |  |  |
|  |  |  |
|  |  |  |
| Result | The program checks that the lesson is not empty and then prints it until the loop has finished with the lesson iterations. | | |
| Outputs | output name | Datatype | Selection condition |
| lessonLearn | String | "Lesson learn number "+(number) +": " + (Lesson) |
|
|
|

| Name or identifier | FR8: Project report with more registered capsules | | |
| --- | --- | --- | --- |
| Summary | The system makes a loop between all the projects, adding their number of registered capsules, until it finds the one with the most capsules that will be printed. | | |
| Inputs | Input name | Datatype | Selection condition |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Result | The system iterates each project modifying the value of the project with the most capsules until it finds the one that has more capsules than the others. | | |
| Outputs | output name | Datatype | Selection condition |
| greaterProject | String | "The project " + (numberProject)+ " have the greator amount of capsules with " + (numberCapsules) + " capsules" |
|
|
|

| Name or identifier | FR9: Report of capsules registered by a collaborator | | |
| --- | --- | --- | --- |
| Summary | The system makes a loop between all the registered capsules looking for a match with the author entered to find out if any capsule has already been made, all the capsules that match the entered person are printed. | | |
| Inputs | Input name | Datatype | Selection condition |
| name | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Result | The system iterates looking for a match between the name entered and the author of a capsule. Any capsule that matches will be printed, if none is printed the system prints that that name has not registered any capsule. | | |
| Outputs | output name | Datatype | Selection condition |
| authorMatchCapsule | String | name + " has create a capsule in project " + numberProyect + ", stage " +numberStage+ ", capsule number "+numberCapsule. |
|
|
|

| Name or identifier | FR10: Search for situations and capsule lessons approved and published in a search string found in the hashtags. | | |
| --- | --- | --- | --- |
| Summary | The system asks the user to enter the keywords to search, the system searches inside the capsules if the keywords match, when it finds a capsule that matches the searched keywords, it checks that the capsule is approved and published to show the user the situation and lesson of that capsule to the user. | | |
| Inputs | Input name | Datatype | Selection condition |
| searchHashtag | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Result | If the system finds a match in the keywords entered by the user and in a capsule that is already published and approved, the situation and the lesson of the capsule will be printed. This process is repeated until no more matches are found for the keywords. | | |
| Outputs | output name | Datatype | Selection condition |
|  |  |  |
|
|
|

| Name or identifier | FR11: Obtain the hashtags of the lessons and situations. | | |
| --- | --- | --- | --- |
| Summary | Within the situations and lessons that the user enters in the capsule, there are some keywords between #Hashtags#, the system detects these words and stores them to be consulted in the future. | | |
| Inputs | Input name | Datatype | Selection condition |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Result | The keywords that are between the hashtags are detected by the system and it stores them for future use for consultation. | | |
| Outputs | output name | Datatype | Selection condition |
|  |  |  |
|
|
|