TABLE OF ANALYSIS AND SPECIFICATION OF REQUIREMENTS

Student : David Artunduaga Penagos

Customer	GreenSQA
User	Collaborators
Functional requirements	● **FR0:** Register project ● **FR1:** Finish stage of a project ● **FR2:** Register capsule ● **FR3:** Approve capsule ● **FR4:** Post Capsule ● **FR5:** Report quantity of capsules by type ● **FR6:** Report lessons learned ● **FR7:** Report project with more registered capsules ● **FR8:** Report capsules by collaborator ● **FR9:** Consult information on published capsules.
Context of the problem	GreenSQA is a company specialized in high-quality software assurance that works on projects for its clients consisting of 6 stages: start, analysis, design, execution, closure, and project monitoring and control. Currently, frequent employee turnover has led to a significant loss of knowledge in the company. To address this situation, the development of software is being planned to allow the recording of project information in a "knowledge capsule," and to also enable its management and consultation. This would effectively preserve the information of each project without relying solely on the individuals who worked on it.
non-functional requirements	● **NRF0:** The software should be quickly and easily usable by the user, with an interface that is easy to understand. ● **NRF1:** The software must be scalable, that is, easily maintainable and upgradable.

<table>
<tr><td>identifier and name</td><td colspan="3">FR0: Register project</td></tr>
<tr><td>Summary</td><td colspan="3">The software must allow the user to register the projects of the company's clients with their respective information. Each stage of the project will be assigned a planned start and finish date. The planned start date is the date the project is created for the “ start ” stage, for the rest of the stages it is the end date of the previous stage. The planned finish date is the date the project is created plus the months that the stages last until the current one. In addition, an actual start and end date must be assigned. The actual start date is the date on which the project is created for the “ start ” stage, for the rest of the stages it is the date on which the previous stage is completed. The actual finish date is the date each stage is completed. In order to establish the planned start dates of the stages, the software must prompt the user for the duration in months of each stage.

When creating a project, the software must create its 6 stages, but only the " start " stage should remain “active”. The software must request and record the following project data: the name of the project, the name of the client, the telephone number of the client, the budget of the project, the name of the project manager and his telephone number.

Additionally, the software must record the planned start date of the project (corresponding to the start date of the " start " stage), as well as the planned finish date of the project (corresponding to the planned finish date of the control stage of the project).</td></tr>
<tr><td rowspan="6">Inputs</td><td>input name</td><td>Datatype</td><td>Condition valid values</td></tr>
<tr><td>projectName</td><td>String</td><td>Characters (Text), no void</td></tr>
<tr><td>clientName</td><td>String</td><td>Characters (Text), no void</td></tr>
<tr><td>clientPhone</td><td>String</td><td>Characters (Text), no void</td></tr>
<tr><td>budget</td><td>double</td><td>Numbers, no void</td></tr>
<tr><td>managerName</td><td>String</td><td>Characters (Text), no void</td></tr>
</table>

	managerPhone	String	Characters (Text), no void
	stageMonths	int[]	Integer numbers
Result or Postcondition	The software should perform the following actions: 1. Save the information provided and create the corresponding project. 2. Establish the stages of the project. 3. Calculate the planned dates of the stages and the actual start date of the “start ” stage. 4. Start stage to "active" and calculate the planned start and finish dates for the project. 5. Issue a message indicating the operation was successful or if an error occurred during registration, and display the information of the project that has been registered.		
Outputs	**output name**	**Datatype**	**Format**
	msg	String	characters(text)
	projectName	String	Characters (Text), no void
	clientName	String	Characters (Text), no void
	clientPhone	String	Characters (Text), no void
	budget	double	Numbers, no void
	managerName	String	Characters (Text), no void
	managerPhone	String	Characters (Text), no void

<table>
<tr><td>identifier and name</td><td colspan="3">FR1: Finish stage of a project</td></tr>
<tr><td>Summary</td><td colspan="3">The software must allow the user to finish each stage of the projects to advance to the next one. To do this, the software must allow recording the approval of the stage and its actual finish date. In addition, the software must update the status of the current stage from “active” to “inactive”, change the status of the new stage from “inactive” to “active”, and record the actual start date of the new stage.</td></tr>
<tr><td rowspan="2">Inputs</td><td>input name</td><td>Datatype</td><td>Condition valid values</td></tr>
<tr><td>searchedProject</td><td>String</td><td>Characters (Text), no void</td></tr>
<tr><td>Result or Postcondition</td><td colspan="3">The software should perform the following actions:

1. Update the status of the current stage to "inactive".
2. Record the actual finish date of the current stage.
3. Record the approval of the current stage.
4. Update the status of the new stage to "active".
5. Record the actual start date of the new stage.

In addition, the software should print a message indicating whether the change was successful or if an error occurred during the stage update.</td></tr>
<tr><td rowspan="2">Outputs</td><td>output name</td><td>Datatype</td><td>Format</td></tr>
<tr><td>msg</td><td>String</td><td>characters(text)</td></tr>
</table>

<table>
<tr><td>identifier and name</td><td colspan="3">FR2: Register capsule</td></tr>
<tr><td>Summary</td><td colspan="3">The software must allow collaborators (who register the capsules) to register the knowledge capsules corresponding to each stage of the projects. To do this, the software must request and register from the capsule: a description of the situation to be registered, a type of capsule (the types defined so far are technical, management, domain and experiences), the name and position of the collaborator , and the learning or lesson learned from the situation. Internally, the software must store a unique identifier for each capsule.

In addition, the software must extract words marked with “#” at the beginning and end (keywords) from the capsule content and store them in a capsule feature called a “hashtag”.</td></tr>
<tr><td rowspan="6">Inputs</td><td>input name</td><td>Datatype</td><td>Condition valid values</td></tr>
<tr><td>description</td><td>String</td><td>Characters (Text), no void</td></tr>
<tr><td>capsuleType</td><td>String</td><td>Technical, management, domain or experiences</td></tr>
<tr><td>collaboratorName</td><td>String</td><td>Characters (Text), no void</td></tr>
<tr><td>collaboratorPosition</td><td>String</td><td>Characters (Text), no void</td></tr>
<tr><td>learning</td><td>String</td><td>Characters (Text), no void</td></tr>
<tr><td>Result or postcondition</td><td colspan="3">The software should perform the following actions:
1. Save the information provided.
2. Create the corresponding capsule.
3. Update capsule status to "under review ".
4. Issue a message indicating whether the operation was successful or if an error occurred during registration and show the capsule information.</td></tr>
</table>

	output name	Datatype	Format
Outputs	msg	String	Characters(text)
	description	String	Characters(text)
	capsuleID	String	Characters(text)
	capsuleType	String	Characters(text)
	collaboratorName	String	Characters(text)
	collaboratorPosition	String	Characters(text)
	learning	String	Characters(text)
	Keywords	String[]	Characters(text)

identifier and name	FR3: Approve capsule		
Summary	The software must allow the user to approve a capsule that can later be published. To do this, the software must request the unique identifier of the capsule, locate it, change the status of the capsule to " approved " and record its approval date.		
Inputs	**input name**	**Datatype**	**Condition valid values**
	capsuleID	String	Numbers or characters, no void
Result or Postcondition	The software should perform the following actions: 1. Perform a search in the registered capsules. 2. Locate the capsule with the name entered. 3. Change the capsule status to " approved ". 4. Record the capsule approval date. 5. Issue a message indicating whether the change was successful or if an error occurred during the process.		
Outputs	**output name**	**Datatype**	**Format**
	msg	String	characters(text)

identifier and name	FR4: Post Capsule		
Summary	The software must allow the user to post a capsule after verifying that it has been approved. The capsules will be generated in HTML format and their URL must be stored. To do this, the software must request the name of the capsule, locate it, change its status to " published ", and store its URL in a capsule feature.		
Inputs	**input name**	**Datatype**	**Condition valid values**
	capsuleID	String	Characters (Text), no void

Result or Postcondition	The software should perform the following actions: 1. Perform a search in the registered capsules. 2. Locate the capsule with the name entered. 3. Change the capsule status to " published ". 4. Store a URL in a capsule feature. 5. Issue a message indicating whether the change was successful or if an error occurred during the process.		
Outputs	**output name**	**Datatype**	**Format**
	msg	String	characters(text)

identifier and name	FR5: Report quantity of capsules for type		
Summary	The software must allow the user to view the number of capsules registered for each type (technical, management, domain and experiences). To do this, the software must search for all registered capsules, count how many there are for each type, and display this information to the user.		
Inputs	**input name**	**Datatype**	**Condition valid values**
	N/A	N/A	N/A
Result or Postcondition	The software should perform the following actions: 1. Search all registered capsules. 2. Count how many there are for each type (technical, management, domain and experiences). 3. Show said information to the user so that they can view the number of capsules registered for each type. 4. Issue a message indicating the operation was successful or if an error occurred during the process.		
Outputs	**output name**	**Datatype**	**Format**
	msg	String	characters(text)
	numCapTechnical	int	Number

	numCapManagement	int	Number
	numCapDomain	int	Number
	numCapExperiences	int	Number

identifier and name	FR6: Report lessons learned		
Summary	The software must allow the user to view a list of the lessons corresponding to the capsules registered in the projects for a particular stage. To achieve this, the software needs to prompt the user for the name of the project and the name of the stage for which the user wants to see their capsule learnings. The software should then search for the project with the given name, locate the corresponding stage, and list the learnings for each of the capsules at that stage.		
Inputs	**input name**	**Datatype**	**Condition valid values**
	projectName	String	Characters (Text), no void
	stageName	String	Characters (Text), no void
Result or Postcondition	The software should perform the following actions: 1. Find the project with the given name. 2. Locate the corresponding stage. 3. List the learning of each of the capsules in that stage. 4. Issue a message indicating the operation was successful or if an error occurred during the process.		
Outputs	**output name**	**Datatype**	**Format**
	learningStage	String[]	characters(text)

identifier and name	FR7: Report project with more registered capsules		
Summary	The software must allow the user to view the project with the highest number of registered capsules compared to the other projects. To achieve this, the software must count the registered capsules for each project, identify the project with the most capsules, and display the corresponding project name to the user.		
Inputs	**input name**	**Datatype**	**Condition valid values**
	N/A	N/A	N/A
Result or Postcondition	The software should perform the following actions: 1. Count the registered capsules for each project. 2. Identify the project with the largest number of capsules. 3. Show the user the name of the corresponding project. 4. Issue a message indicating the operation was successful or if an error occurred during the process.		
Outputs	**output name**	**Datatype**	**Format**
	msg	String	characters(text)
	projectName	String	Characters (Text)

identifier and name	FR8: Report capsules by collaborator		
Summary	The software must allow the user to view the number of capsules with their IDs that a specific collaborator has registered. To do this, the software must ask the user for the name of the collaborator and search the registered capsules for those that match the name provided, extract their IDs and display them to the user. If there is no capsule registered by the indicated collaborator, the software should inform the user by means of a message.		
Inputs	**input name**	**Datatype**	**Condition valid values**

	collaboratorName	String	Characters (Text), no void
Result or Postcondition	The software should perform the following actions: 1. Search for the registered capsules that match the collaborator's name. 2. Extract the IDs of the capsules found. 3. Show the capsule IDs to the user. 4. Inform the user if no capsule registered by the collaborator was found. 5. Indicate to the user by means of a message if the operation was carried out successfully or if an error occurred.		
Outputs	**output name**	**Datatype**	**Format**
	msg	String	characters(text)
	collaboratorCaps	String[]	characters(text)

identifier and name	FR9: Consult information on published capsules.		
Summary	The software must allow the user to view the learnings from the approved and published capsules through a search function based on the hashtags. To achieve this, the software must prompt the user to enter the search text and search for it in the hashtags of the approved and published capsules. The software should then show the user the capsule learnings that match the search criteria. If there are no approved or published capsules matching the search criteria, the software should notify the user via a message.		
Inputs	**input name**	**Datatype**	**Condition valid values**
	text	String	Characters (Text), no void
Result or Postcondition	The software should perform the following actions: 1. Search the hashtags of the approved and published capsules. 2. Show the user the learnings of the capsules that match the search criteria. 3. Inform the user if no registered capsule was found that matches the search criteria.		

	4. Indicate to the user by means of a message if the operation was carried out successfully or if an error occurred.		
Outputs	**output name**	**Datatype**	**Format**
	msg	String	characters(text)
	learnings	String[]	characters(text)