Language Detection Web Application – Duck Soft Works § Co.

US 3 - As a user, I want to obtain information about the tasks in progress and already completed, by category and/or status.

1. Requirements Engineering

1.1. Customer Specifications and Clarifications

N/A - No questions have been asked in order to further clarify requirements for this US.

1.2 Customer Specifications and Clarifications

From the specifications document:

The system should implement functionalities that allow users to retrieve information for tasks by state of completion , category or both.

1.3. Acceptance Criteria

If no tasks have been submitted at all, no results should be found.

Tasks retrieved in the search must only be those submitted by the user.

No tasks should be retrieved , when searching by category , if no task was previously submitted with said category.

When filtering by state of completion, no tasks besides the ones with the specified state of completion should be shown.

1.4. Found out Dependencies

Dependency with US1, since task creation must exist prior to task search being possible

1.5 Input and Output Data

To find a task by category , a valid category name must be provided.

To find a task by status of completion , a valid status must be provided.

Output data will always be based on the kind of filters applied in the request , returning an appropriate response showing information relevant to the request.

On the event that there are no tasks found for a specific search (be it with a category and/or satus filter or by having no filters) no information is found.

If data introduction for a request cannot be processed by the system, the user will be notified on the client side appropriately.

The data shown is user specific. The user can only fetch information for tasks which he/she submitted.
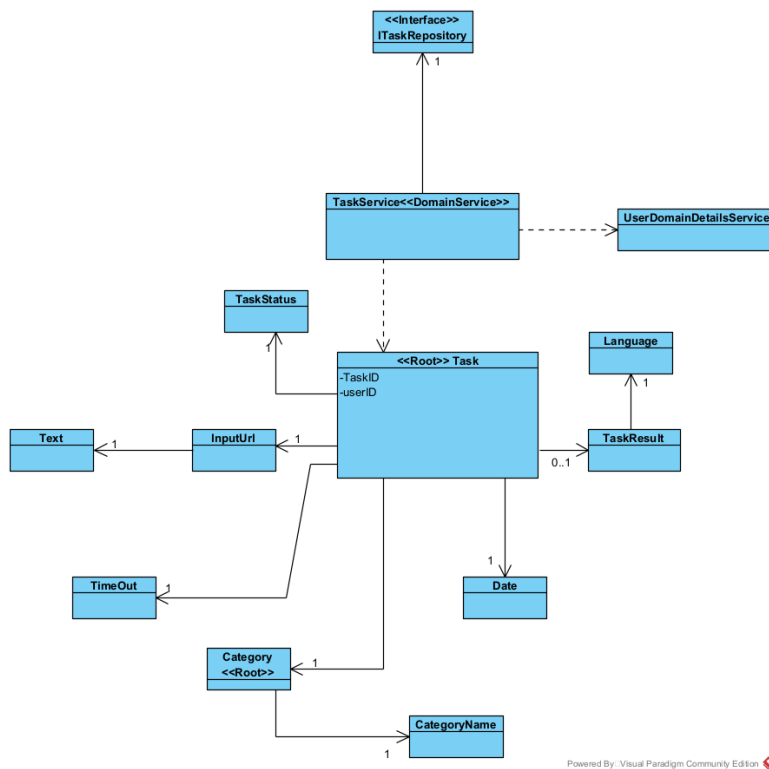
1.6. System Sequence Diagram (SSD)
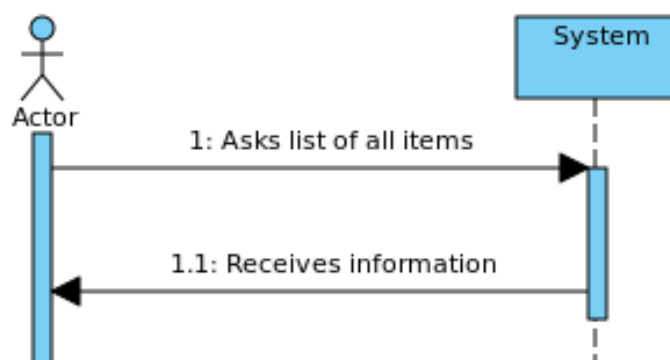
This SSD illustrates the actions performed by the user actor. Those actions can be to find all tasks, find all tasks by category, find all tasks by status and find all tasks by status and category.

2. OO Analysis

### 2.1. Relevant Domain Model Excerpt

An excerpt of the general domain model, adapted for this US. The represented concepts are the Task and it's value objects, the TaskService as the domain service that uses the UserDomainDetailsService to fetch user specific information and uses functionalities for database querying with ITaskRepository.

3 Sequence Diagram (SD)

For these sequence diagrams, and because it involves actions that are very similar to other User Stories for the application, please check the Generic FindAll SD that is present in the team's report.

3.1 Class Diagram (CD)
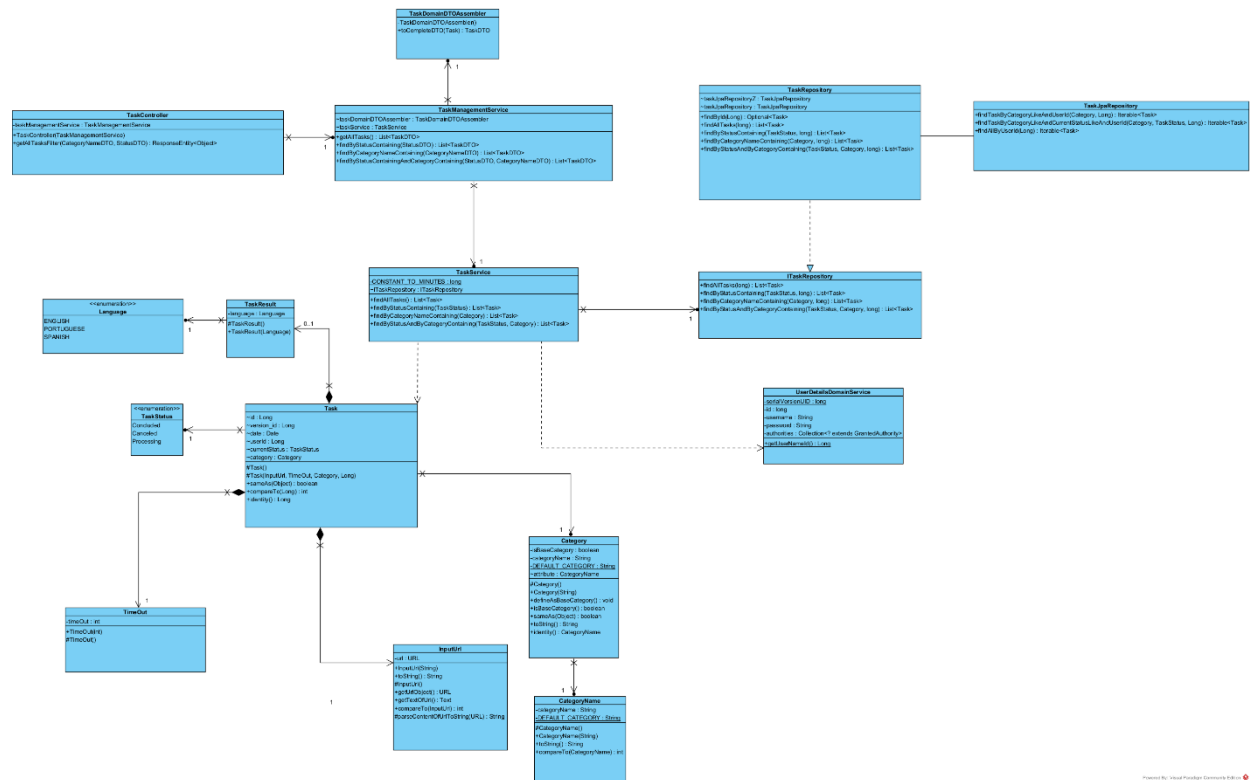
A class diagram excerpt representing all of the classes, relevant attributes and methods that play a part on user task information fetching functionalities.

4. Tests

This is a mere example of one of the tests performed for this use case.

This in particular is for a retrieving a user's specific tasks when searching by CategoryName and Status.

```
@Test
void ensureFindTaskByCategoryNameAndStatusReturnListOfTasksOfCorrespondingUser(){
    Category category = new Category( categoryDescription: "Sports");
    Task task = Mockito.mock(Task.class);
    try(MockedStatic<UserDetailsDomainService> utils = Mockito.mockStatic(UserDetailsDomainService.class)){
        utils.when(()-> iTaskRepository.findByStatusAndByCategoryContaining(any(TaskStatus.class), any(Category.class), any(Long.class))).thenReturn(List.of(task));


        assertEquals(taskService.findByStatusAndByCategoryContaining(TaskStatus.Concluded, category), List.of(task));
    }
}
```

5. Construction (Implementation)

Implementation was based on the basis that searching a task could be both by category or state. We also provided methods that allow searching by both .

Even though there isn't a specific requirement to find all tasks, regardless of criteria for either category or state, this functionality was implemented as we considered that it could prove itself to be useful in terms of scalability.

Communication between the domain service and the repository are achieved by using na interface that abstracts the database transaction implementation so that different possibilities of persistence are possible , in the future.

Finding a user specific task , rules out possibilities for fetching information for tasks relative to a different user. Likewise, an administrator does not have the possibility of fetching this information either.

6. Integration and Demo

Task searching related functionalities require a user to be logged in with regular user permissions, so that the functionalities for this US can be available for he/she to use.

We made an integration with one frontend application that can be used in conjunction with the backend application.

Ducked Language Detection

Check your Tasks

| Categories | All | Status | All | Check |

Your Tasks

Task ID	Date	Text Category	URL	Text Language	Task Status	Time
22	17:00:19 28-06-2022	Economics	file:/C:/Users/thale/Downloads/example (1).txt	ENGLISH	Concluded	1
21	17:00:11 28-06-2022	Sports	file:/C:/Users/thale/Downloads/example (1).txt	ENGLISH	Concluded	1
20	16:59:58 28-06-2022	Philosophy	file:/C:/Users/thale/Downloads/example (1).txt	ENGLISH	Concluded	1
19	16:48:34 28-06-2022	Philosophy	https://www.w3.org/TR/PNG/iso_8859-1.txt	ENGLISH	Concluded	1
18	16:48:34 28-06-2022	Philosophy	https://www.w3.org/TR/PNG/iso_8859-1.txt	ENGLISH	Concluded	1
17	16:48:34 28-06-2022	Philosophy	https://www.w3.org/TR/PNG/iso_8859-1.txt		Canceled	1
16	16:48:31 28-06-2022	Philosophy	https://www.w3.org/TR/PNG/iso_8859-1.txt	ENGLISH	Concluded	1
15	16:04:18 28-06-2022	Philosophy	https://www.w3.org/TR/PNG/iso_8859-1.txt	ENGLISH	Concluded	1
14	15:57:02 28-06-2022	Philosophy	https://www.w3.org/TR/PNG/iso_8859-1.txt	ENGLISH	Concluded	1

Ducked Language Detection

Check your Tasks

| Categories | Economics | Status | All | Check |

Your Tasks

Task ID	Date	Text Category	URL	Text Language	Task Status	Time
22	17:00:19 28-06-2022	Economics	file:/C:/Users/thale/Downloads/example (1).txt	ENGLISH	Concluded	1

7. Observations

The US realization was achieved with success.

Some aspects worth mentioning :

- On the event that no tasks can be fetched when a user search is initiated, the dev team opted to respond with an empty list. This was due to deeming that an empty list is still a success case. This, in turn, allows the client side to deal with this information as it sees fit.

-As far as the UI is concerned, the way to discern between the user's created tasks are the task id and timestamp. There were no requirements on if the tasks had to be sorted/ordered by specific criteria. Despite this, we believe the presented data is able to give some insight on which task is which and that it should allow the user to be knowledgeable of this.