Technical Report

- Andres Jose Acevedo Avila
- Javier Camilo Murcia Nahan

Diagrams:

- CRC Cards

User	
- Verify what the customer can access	Client

Authetication	
- Log in - Log out	Client

SubTask	
-Store and manage subtasks -Add, get, and remove subtask	Component

Notification	
- Send and show notification -Store the message	Client

Client	
- Create, edit, view and delete tasks - Create, edit, view and delete subtasks - Create, view and delete tag - Create, edit, view and delete project - Receive notifications - Start, stop and custom pomodoro timer - See plans - Pay for subscription - Get and create report premium clients -Get and create report clients	User Autenthication Task Subtask Tag Project Notification Pomodoro Plan Subscription

Component	
- Add, obtain and remove components - Store the name and status of the component - Mark the component as done	Subtask Task Project Folder Client

Task	
-Store and manage tasks -Add, get, and remove task -Mark the task as done	Component

Pomodoro	
- Store short and long break duration - Store pomodoro length -Store number of Pomodoros before a long break	Client

Folder	
- Store folder name -Manage projects -Add, get, and remove projects	Component Client

Projects	
-Store and manage projects -Add, get, and remove projects -Mark the project as done	Component

Plan	
-Store plan name, price and description	Client

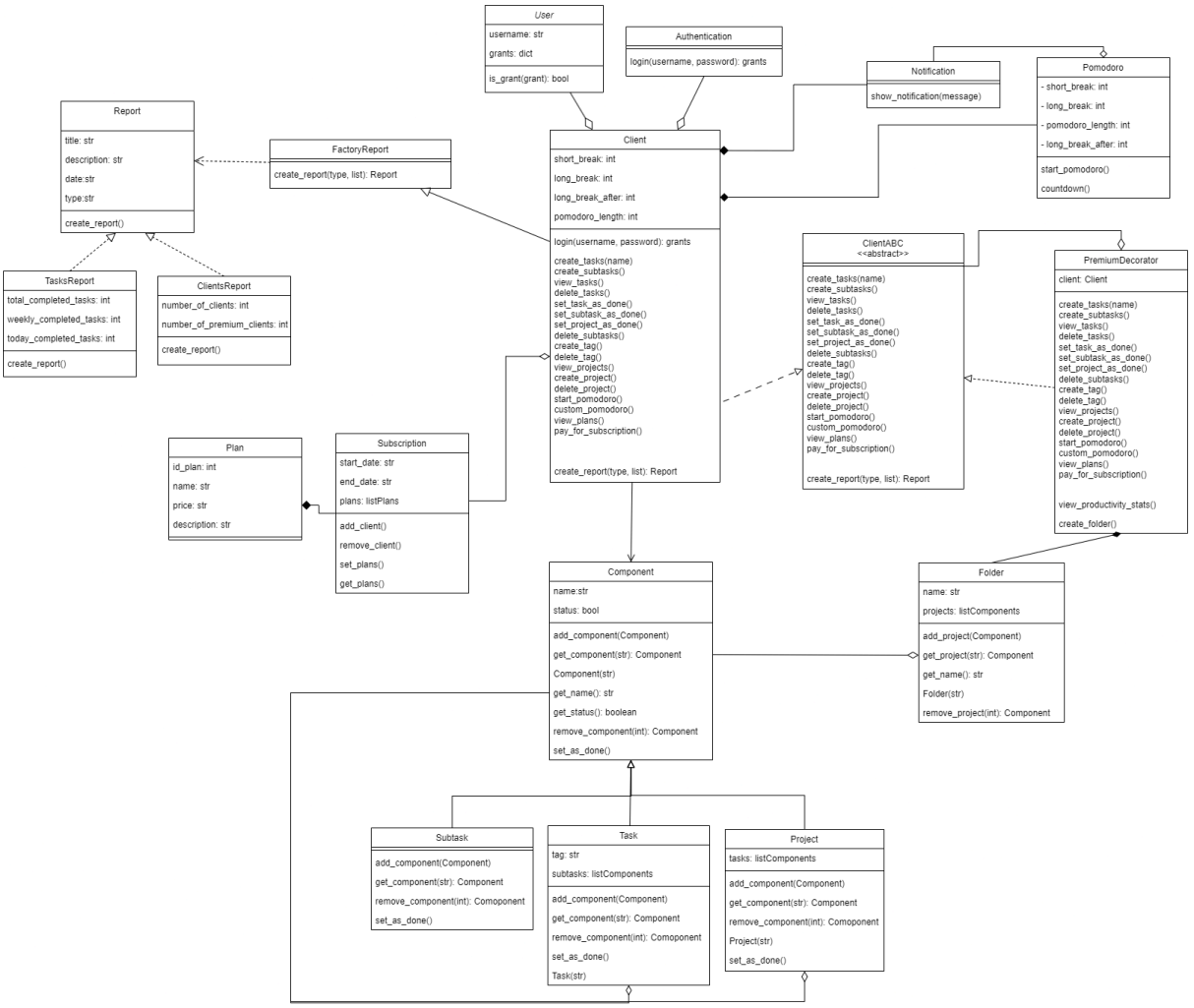
Report	
-Manage basic report information: title, description, date, and type -Provide methods to fill in basic information -Create the report	Client

TasksReport	
- Manage specific task information such as total completed tasks, weekly completed tasks, today completed tasks, task focus time, and project time	Report

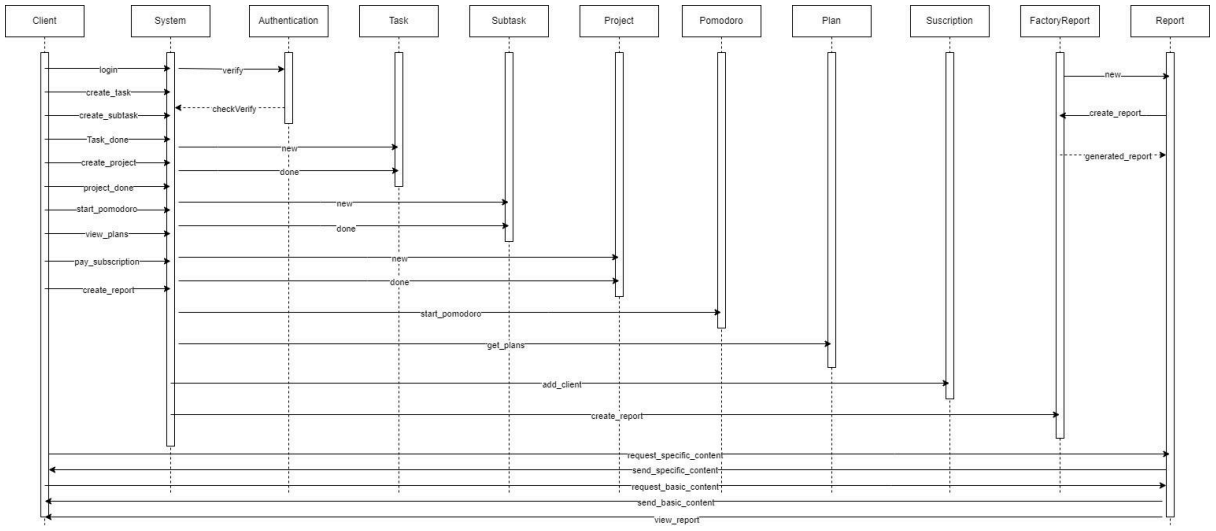
ClientsReport	
- Manage specific client information such as the number of clients and the number of premium clients - Specific report	Report

Subscription	
-Store start date and finish date -Add and remove a client -Modify plan	Client Plan

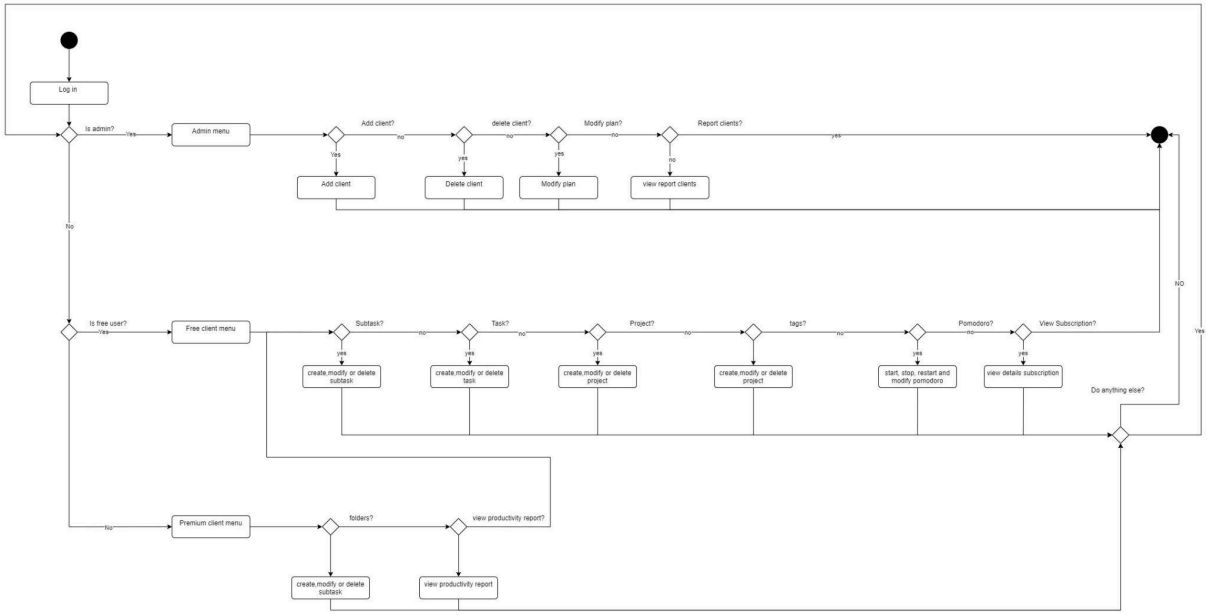
- **Class Diagram**



- **Sequence Diagram**



- **Activity Diagram:**



Technical Decisions:
As end users, only one client was defined, and it was planned to use authentication permissions to designate the admin, as they only had one function: generating client reports.

The template pattern was initially considered for generating the report, since there was a static part (general report data) and a dynamic part depending on the type of object (specific report data). However, it was realized that the report follows the same structure with different content depending on the report type, so the factory pattern was used to allow report generation from the child classes.

The composite pattern was utilized for the creation of subtasks, tasks, and projects. The creation of folders was not considered within the composite, as they are not contained by anything. The decorator pattern was planned to define additional functions for premium clients.

The proxy pattern could have been used to handle the logging of completed tasks as a cache, but it would have added unnecessary complexity to the program, given that simply sending a list of completed tasks was sufficient.

As design patterns were utilized, the SOLID principles are reflected, and spaghetti code was avoided, as each class had its own responsibility. With the factory pattern, we see the Liskov substitution principle in action, and the main implementation is improved, allowing for openness to extension and closure to modification.

Considerations:

This version is incomplete due to poor time management by the developers. So the implementation of web services, and the database are still missing. The developers learned that they need to better organize their time management and not take it for granted.