ENGENHARIA DE SOFTWARE

2022/2023

Projeto - Fase 2

Autores:

45345 - Inês Carvalho

54622 - Tiago Coelho

58013 - Rodrigo Lopes

59208 - Simão Carrasco

60597 - Pedro Catarino

Use Case Diagrams & Descriptions

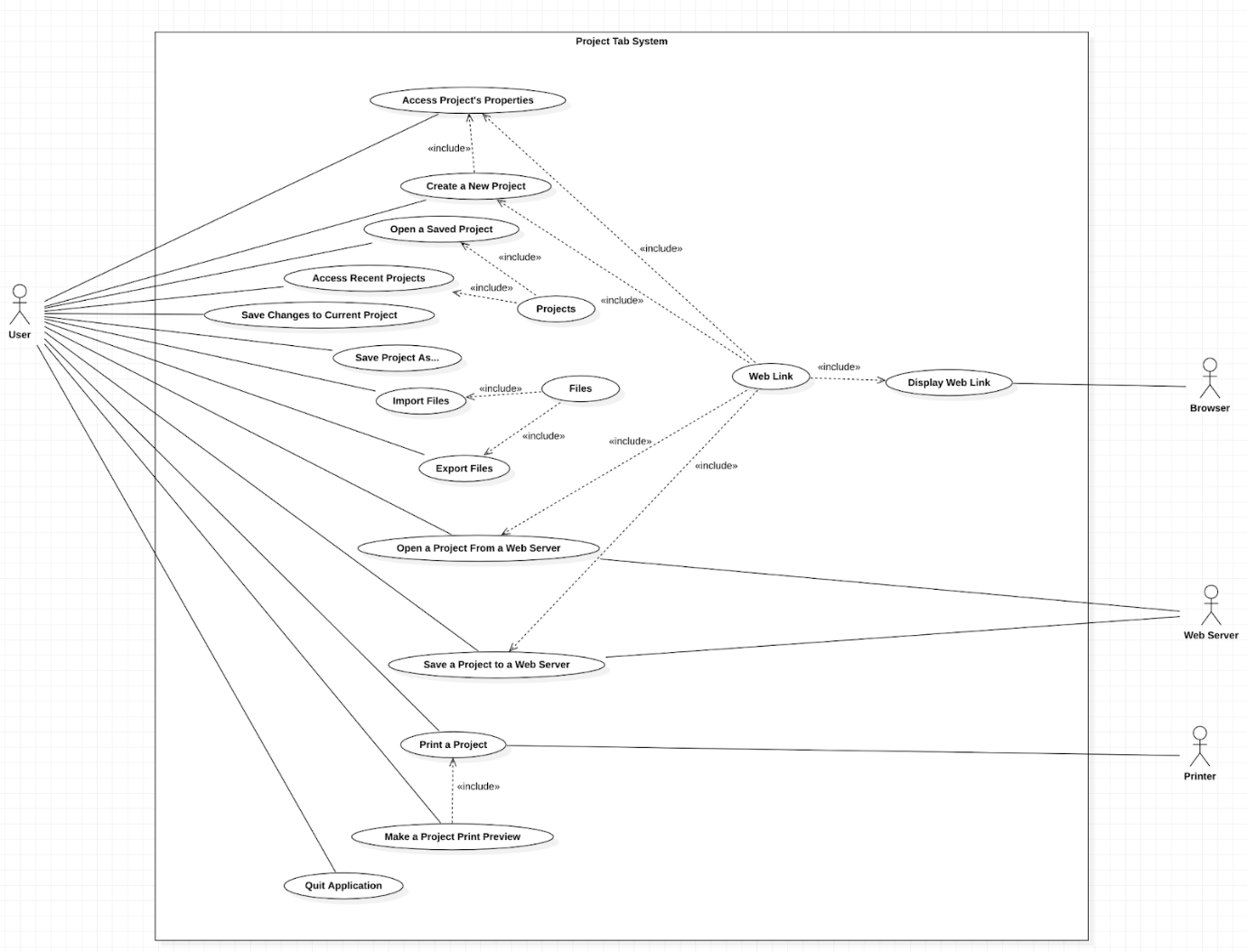
* 45345 - Inês Carvalho:

·      Name: Project Tab system

·      Description: Project tab to be able to manage projects. By clicking on this tab several options should appear (Properties, New, Open, Recent Projects, Save, Save As, Import, Export, Web Server, Print, Print Preview, Quit)

·      Actors:

§  Main: User

****

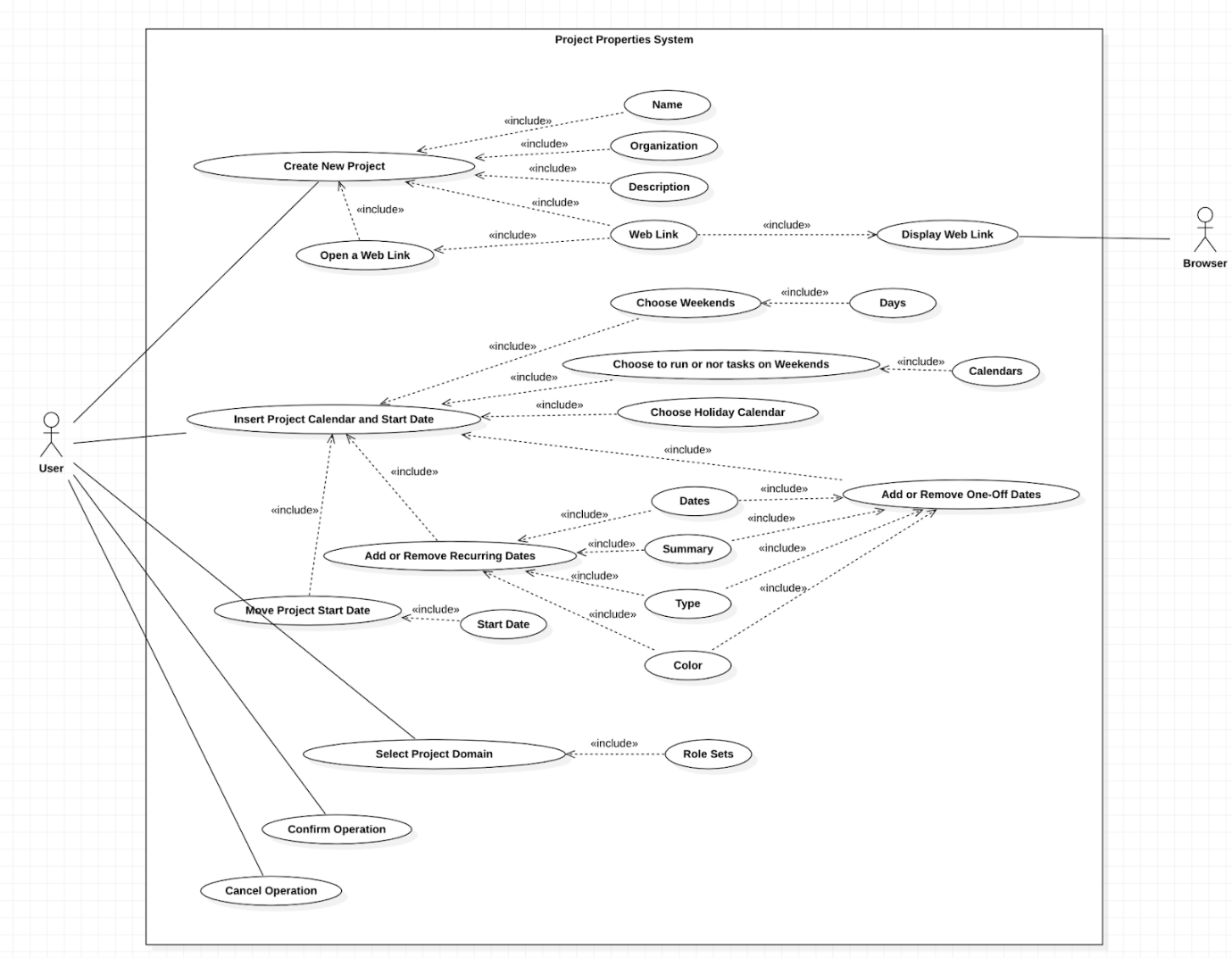
This use cases are related to managing a project in general. They relate to the Project tab on the app’s menu mentioned above.

·      Name: Project Properties System

·      Description: Project properties option. By clicking on this option, a pop-up menu should appear to manage the current projects’ properties

·      Actors:

§  Main: User

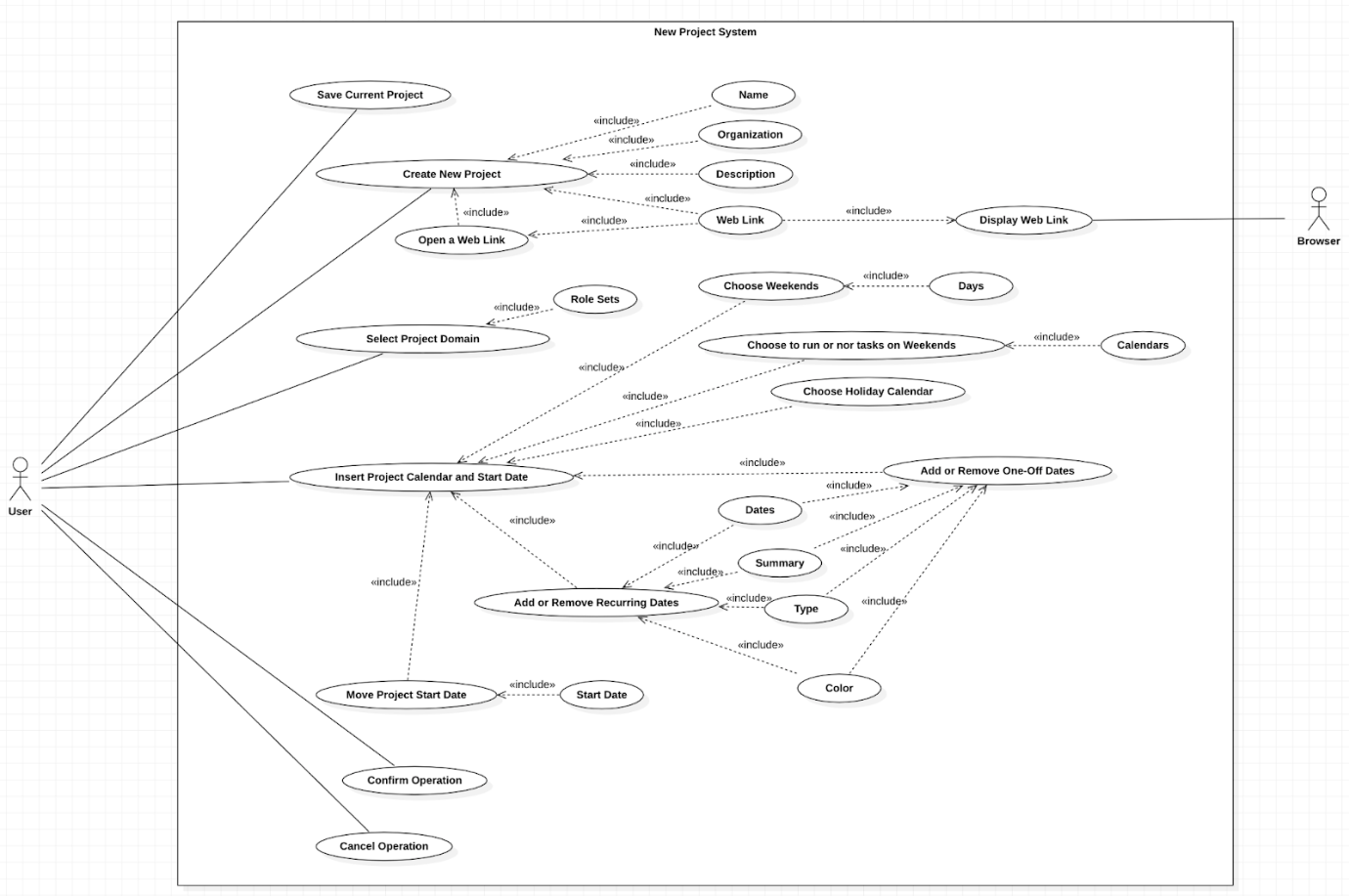


·      Name: New Project System

·      Description: Create a new project option within 3 steps. By clicking this option, a pop-up menu should appear allowing to add some necessary data regarding the project

·      Actors:

§  Main: User

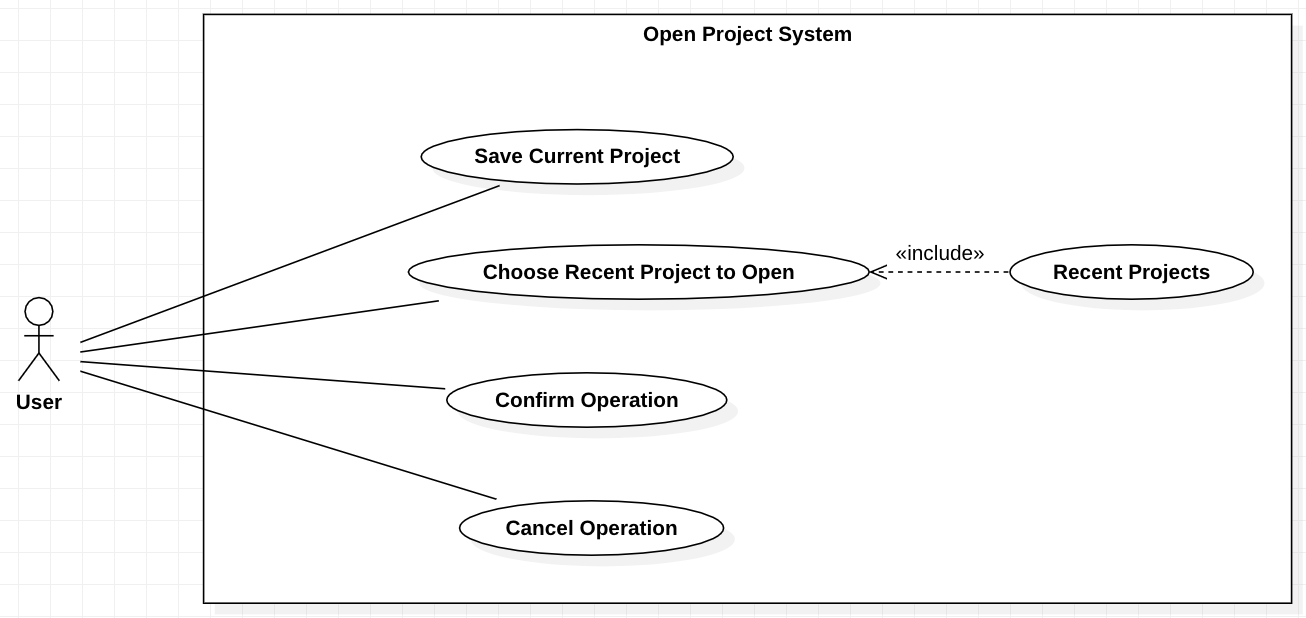


·      Name: Open Project System

·      Description: Open a project option. By clicking this option, a pop-up menu should appear allowing to choose projects from the local file system.

·      Actors:

§  Main: User

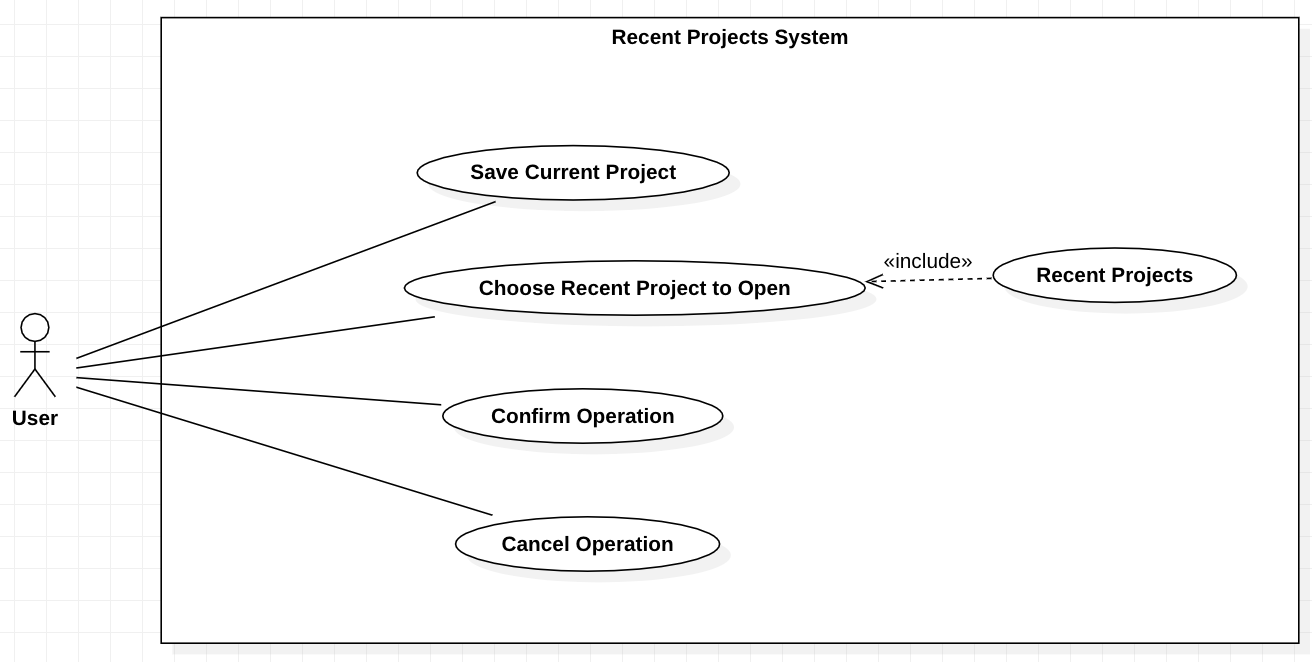


·      Name: Recent Projects System

·      Description: Recent projects option. By hovering over this option, a list of recently opened projects should appear to have a quicker way to open them

·      Actors:

§  Main: User

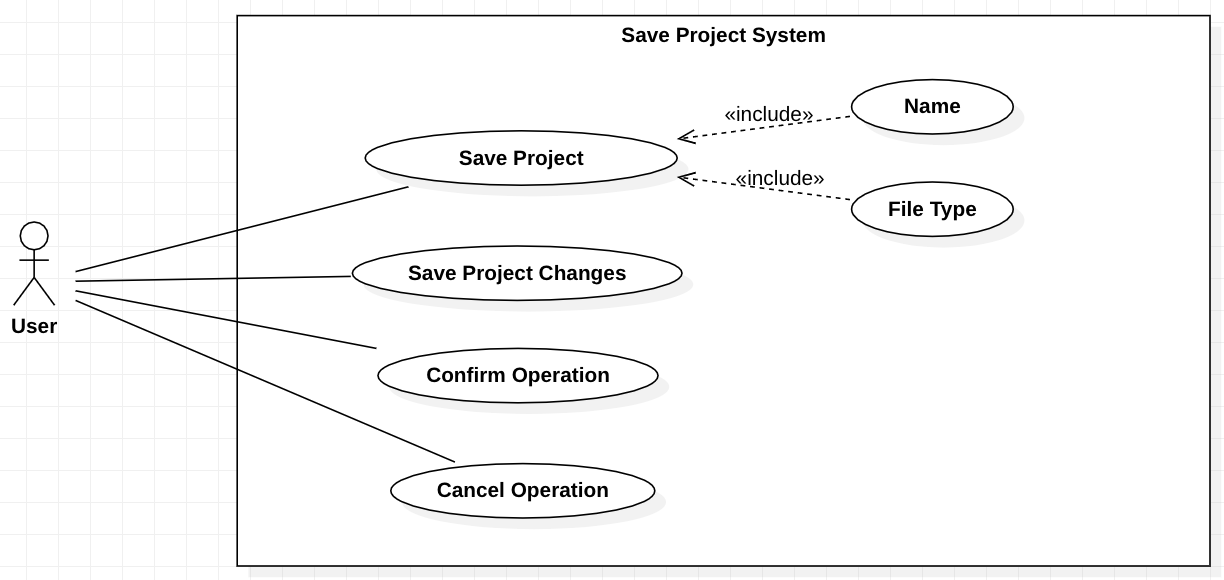


·      Name: Save Project System

·      Description: Save project option. By clicking in this option, an opened project that was already previously saved in the local file system, should make changes to it, otherwise it should behave like the Save As Project option

·      Actors:

§  Main: User

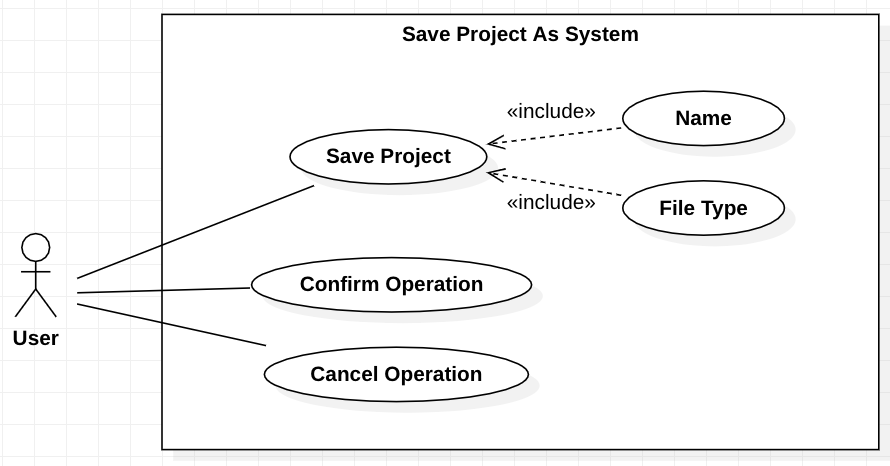


·      Name: Save Project As System

·      Description: Save as project option. By clicking in this option, a pop-up menu should appear allowing to save the file in the local file system

·      Actors:

§  Main: User



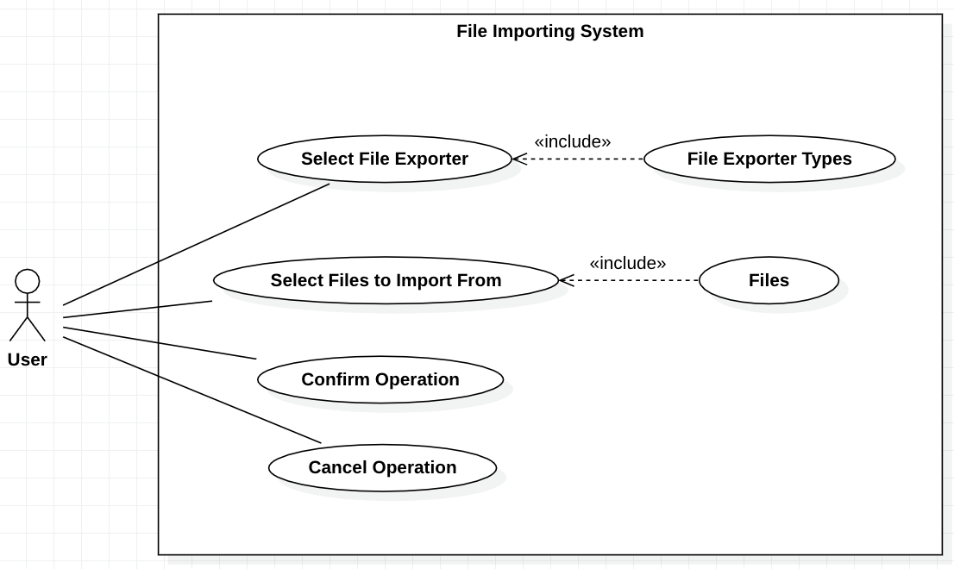
·      Name: File Importing System

·      Description: Select a file importer option. By clicking in this option, a pop-up menu should appear allowing to choose between several file importers (iCalendar file, Microsoft Projects files, GanttProject files, Text files, Comma-separated values files)

·      Actors:

§  Main: User

§  Secondary: File importer



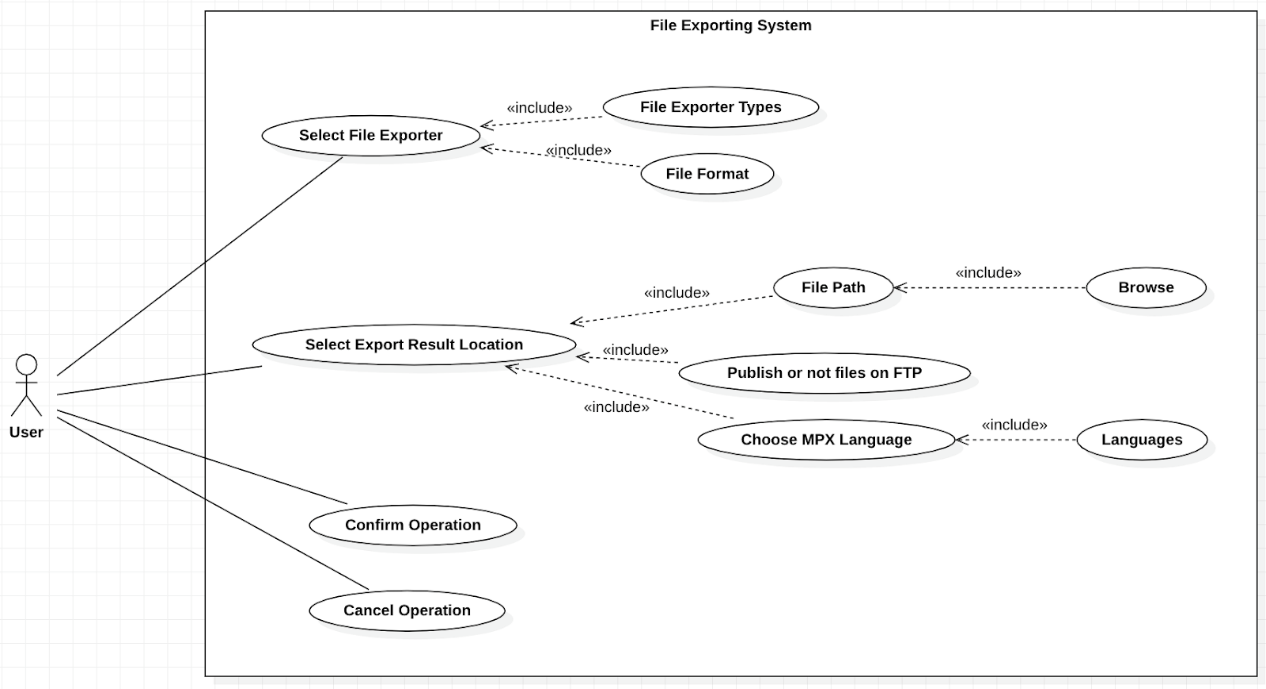
·      Name: File Exporting System

·      Description: Select a file exporter option within 2 steps to import a file. By clicking in this option, a pop-up menu should appear allowing to choose between several file exporters (Microsoft Project file, Raster image file, Comma-Separated-Values file, HTML report, PDF report)

·      Actors:

§  Main: User

§  Secondary: File exporter



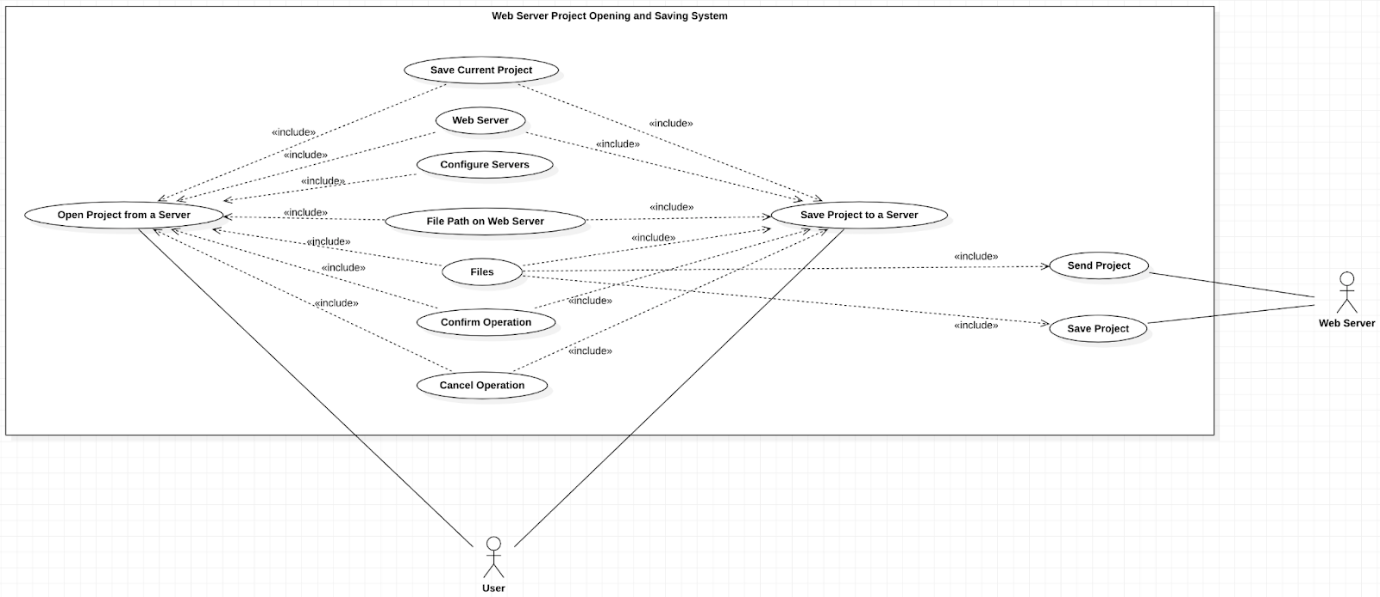
·      Name: Web Server Project Opening/Saving System

·      Description: Web server option. By hovering over this option, the options of opening/saving a project from/to a life server

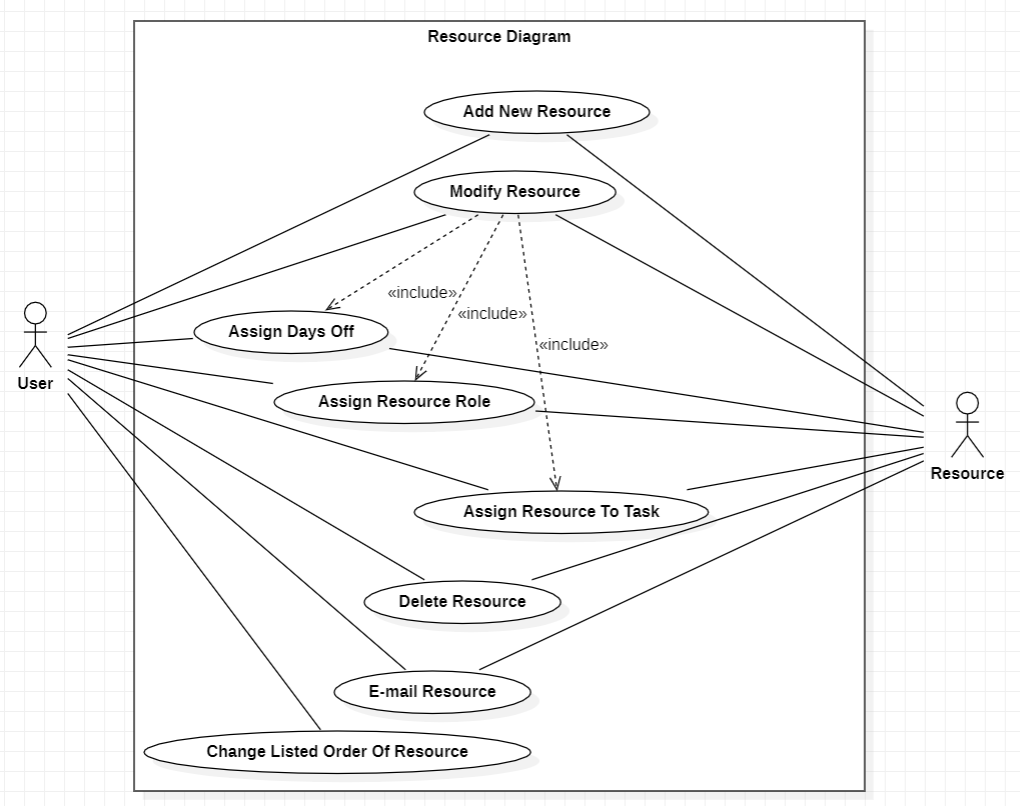
·      Actors:

§  Main: User

§  Secondary: Web Server



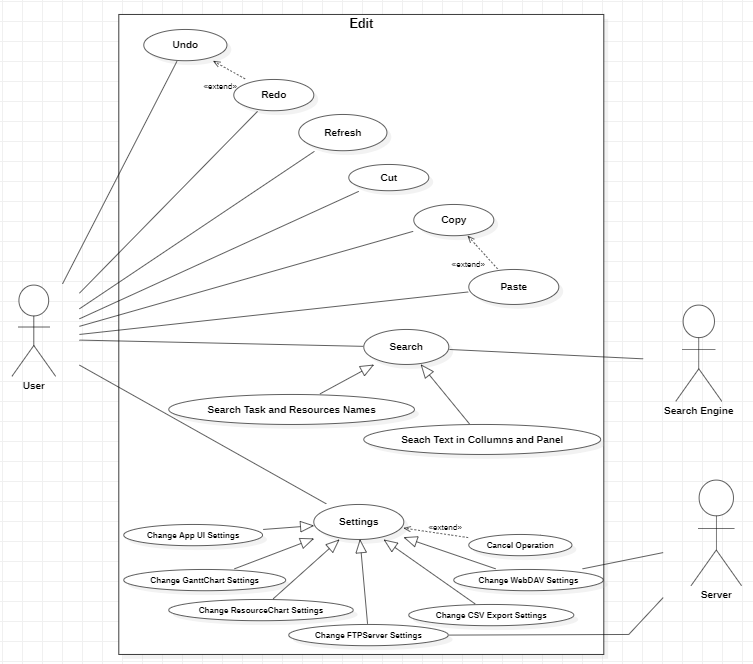
* 58013 – Rodrigo Lopes:
* 59208 – Simão Carrasco:



|  |
| --- |
| Actors:  **User**: The main user that interacts with the application.  **Resource**: A resource symbolizes a person who completes a task, but it can also mean other things, like equipment or materials.  Use Cases:  **Add New Resource**: The user adds a new resource, in the resource diagram.  Primary Actor: User   Secondary Actor: Resource  **Modify Resource**: The user modifies a resource, in the resource diagram.  Primary Actor: User   Secondary Actor: Resource  **Assign Days Off**: The user assigns the days off of a resource, in the resource diagram.  Primary Actor: User   Secondary Actor: Resource  **Assign Resource Role**: The user assigns a resource role to a resource, in the resource diagram.  Primary Actor: User   Secondary Actor: Resource  **Assign Resource To Task**: The user assigns a resource to an existing task, in the resource diagram.  Primary Actor: User   Secondary Actor: Resource  **Delete Resource**: The user deletes a resource, in the resource diagram.  Primary Actor: User   Secondary Actor: Resource  **E-mail Resource**: The user sends an email to a resource, in the resource diagram.  Primary Actor: User   Secondary Actor: Resource  **Change Listed Order Of Resource**: The user changes the listed order of a resource, in the resource diagram.  Primary Actor: User |

* 60597 – Pedro Catarino:

Use Case Diagram 1 – Edit Tab



**Name:** Undo

**Description:** Undoes the most recent operation.

**Main actor:** User

**Name:** Redo

**Description:** Remakes the operation that was undone by the undo action. Can only be used if there was an undo action before.

**Main actor:** User

**Name:** Refresh

**Description:** Updates the application, looking for any new changes.

**Main actor:** User

**Name:** Search

**Description:** Searches for tasks and resources in a project using one of two methods. One method searches task and resource names and task notes, and the other searches text in the columns in the task or resource panel.

**Main actor:** User

**Secondary actors:** Search engine

**Name:** Cut

**Description:** Removes the selected item.

**Main actor:** User

**Name:** Copy

**Description:** Copies the selected item into the clipboard.

**Main actor:** User

**Name:** Paste

**Description:** Pastes what is in the clipboard. Only possible if there was a copy action before.

**Main actor:** User

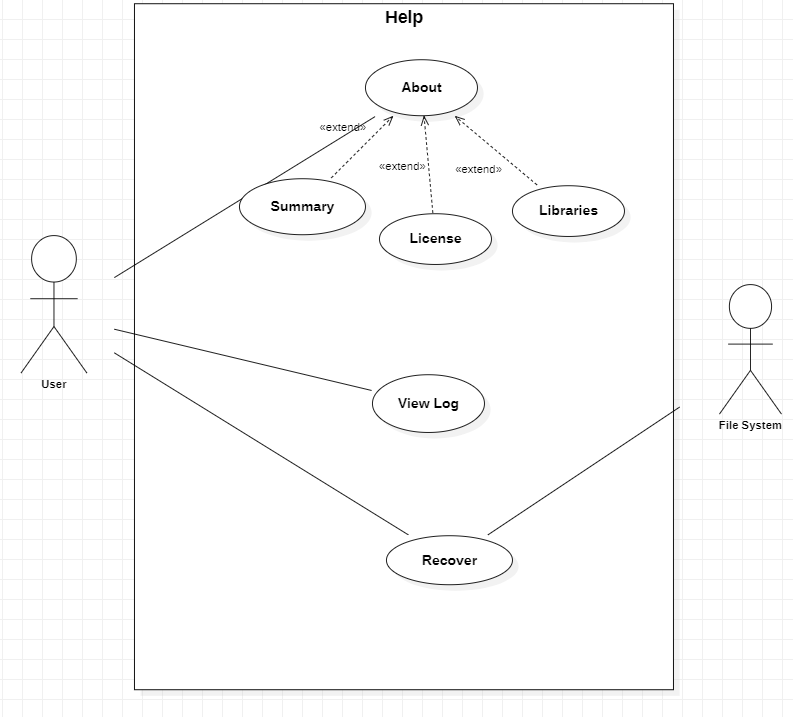
**Name:** Settings

**Description:** Opens a pop-up with the settings options, offers the possibility to change the settings of different components (Application UI, Gantt Chart, Resource Chart, FTP server, CSV export and WebDAV server)

**Main actor:** User

**Secondary actors:** Server

Use Case Diagram 2 – Help Tab



**Name:** About

**Description:** Shows information on the GanttProject version and development team, the GanttProject software license, and the software libraries and versions used in GanttProject.

**Main actor:** User

**Name:** View Log

**Description:** Contains error messages and other information.

**Main actor:** User

**Name:** Recover

**Description:**  Retrieves an earlier version of the project file.

**Main actor:** User

**Secondary actor**: File system

**Name and Number:** Inês Carvalho 45345

**Review:** Use cases have a direct name with appropriate descriptions. Actors correctly identified.

Codebase Metrics Assessment

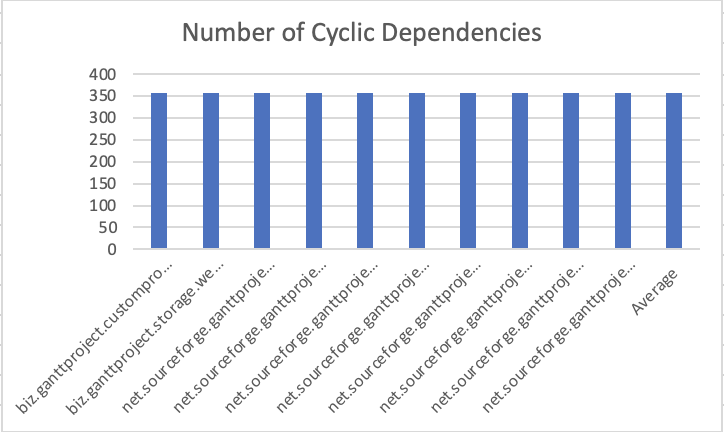
* 45345 - Inês Carvalho:

**Metrics Set: Dependency Metrics**

* Class Metrics:

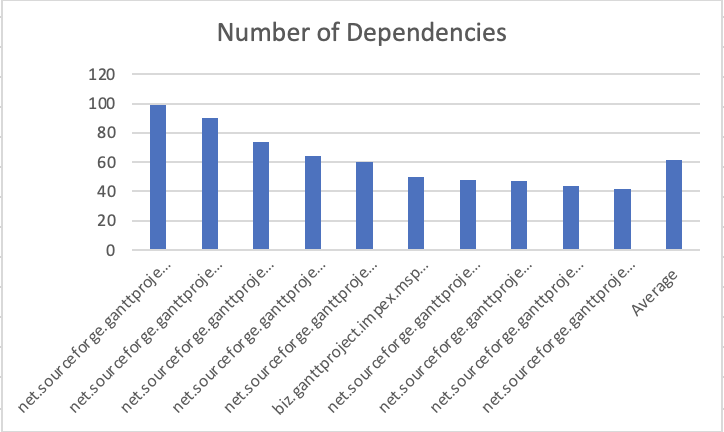
Uma imagem com mesa

Descrição gerada automaticamente



Uma imagem com mesa

Descrição gerada automaticamente



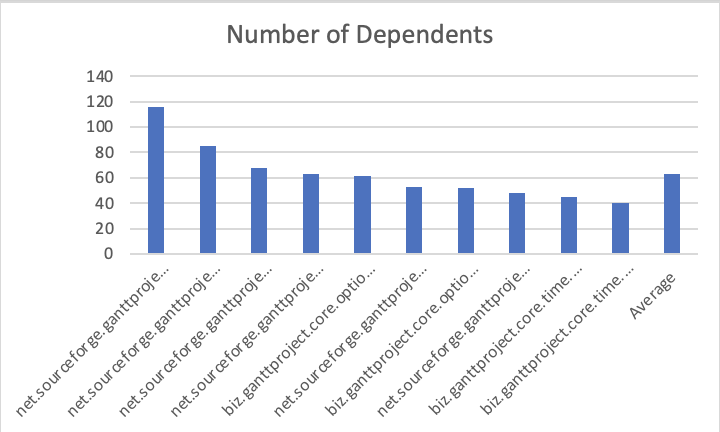
Uma imagem com mesa

Descrição gerada automaticamente



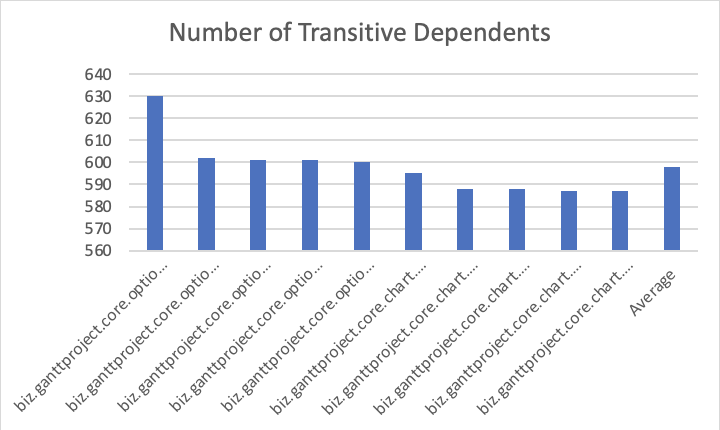
Uma imagem com mesa

Descrição gerada automaticamente



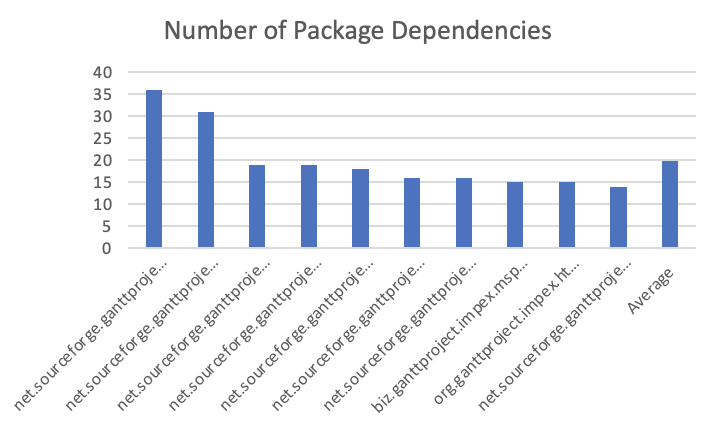
Uma imagem com mesa

Descrição gerada automaticamente



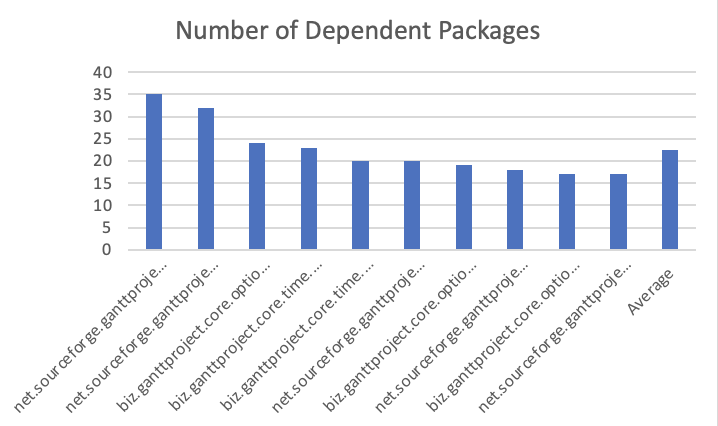
Uma imagem com mesa

Descrição gerada automaticamente



Uma imagem com mesa

Descrição gerada automaticamente

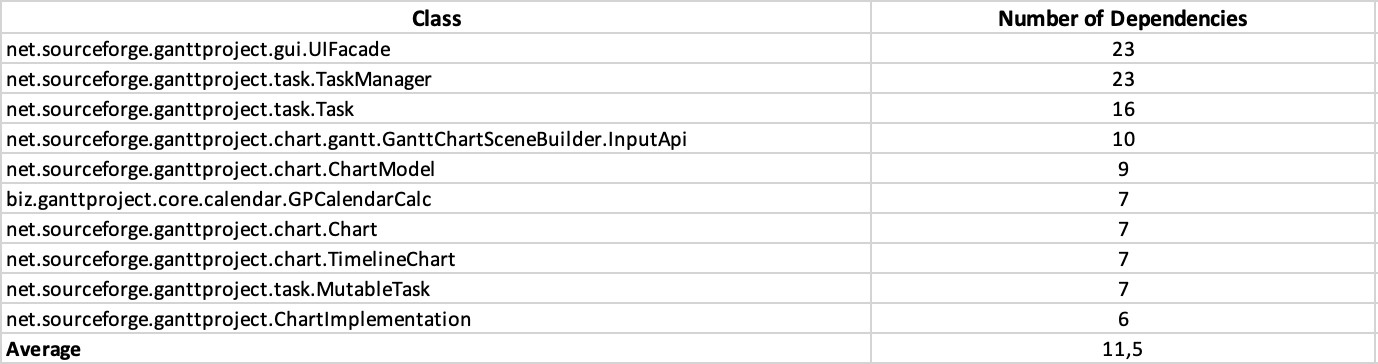


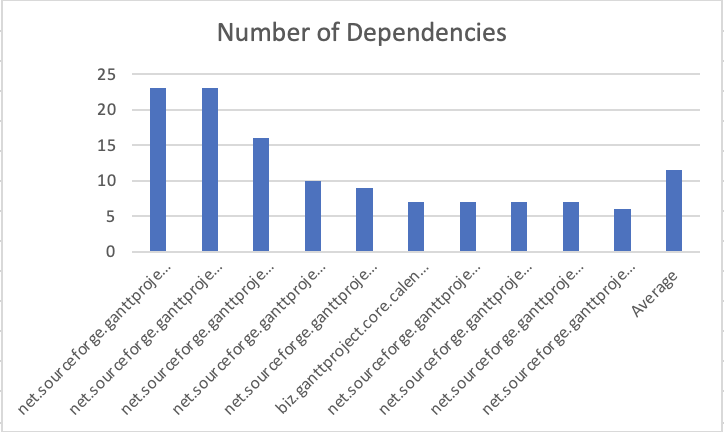
* Interface Metrics:

Uma imagem com texto

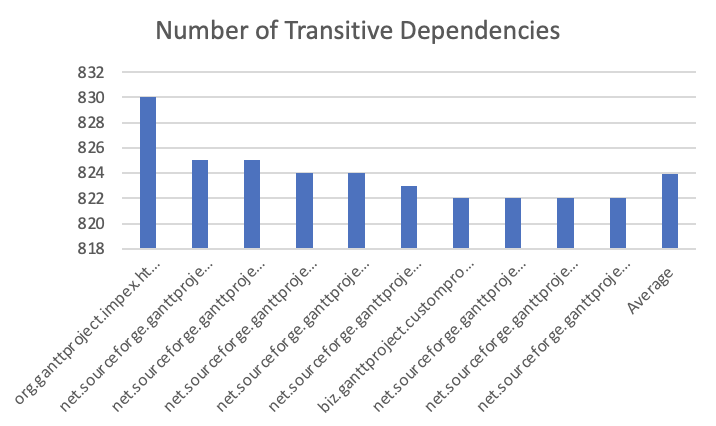
Descrição gerada automaticamente



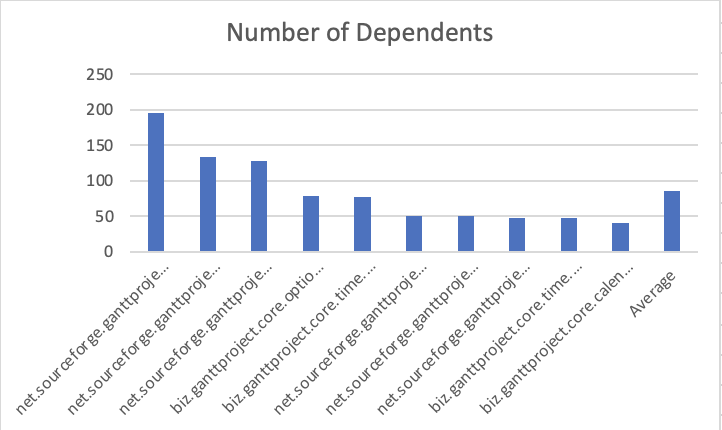




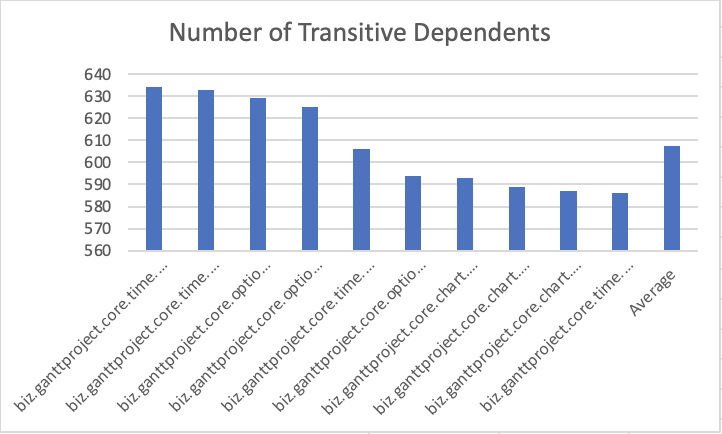


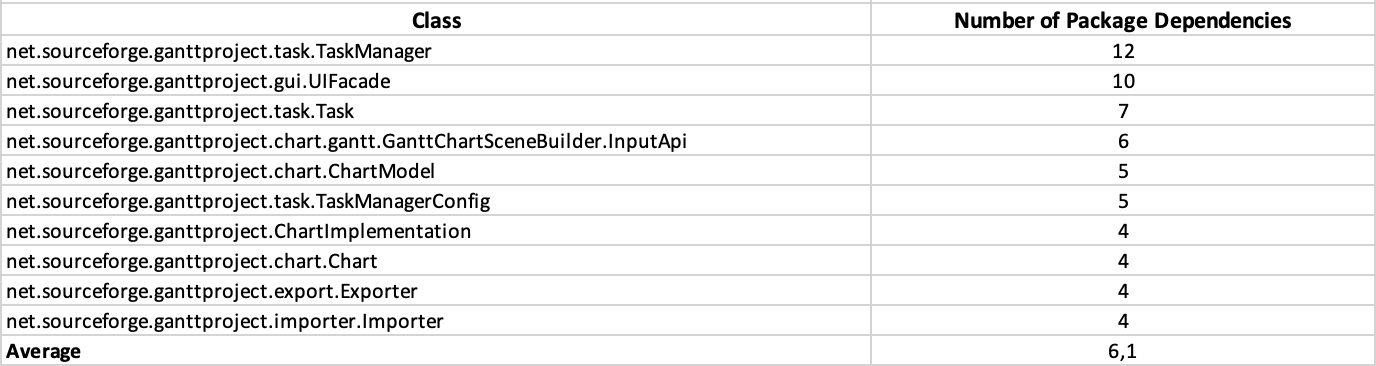


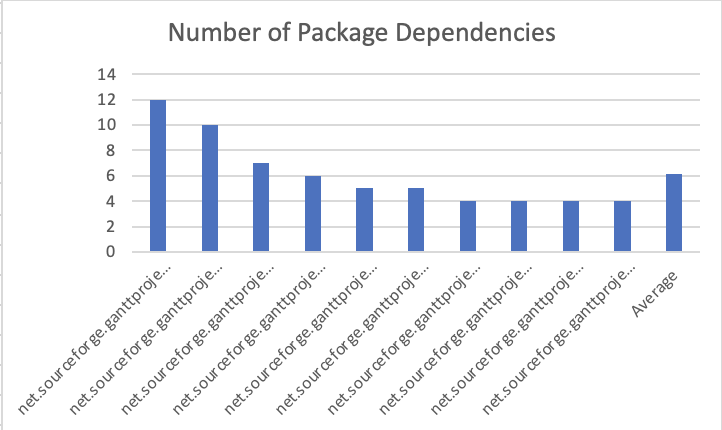




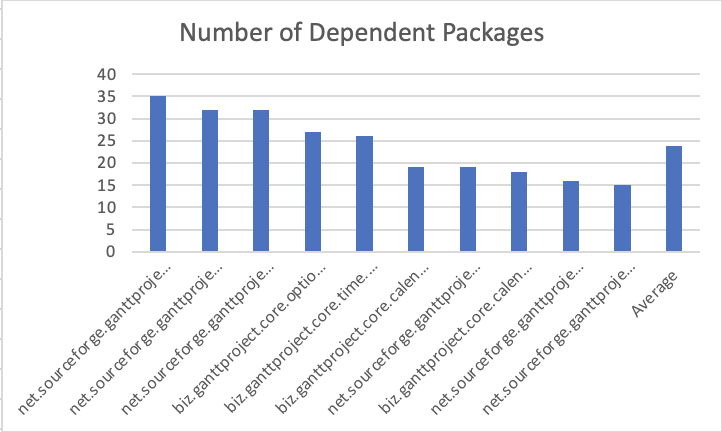








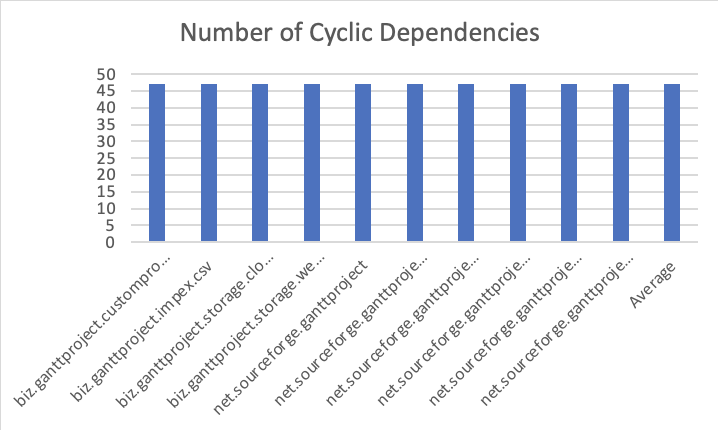




* Package Merics:

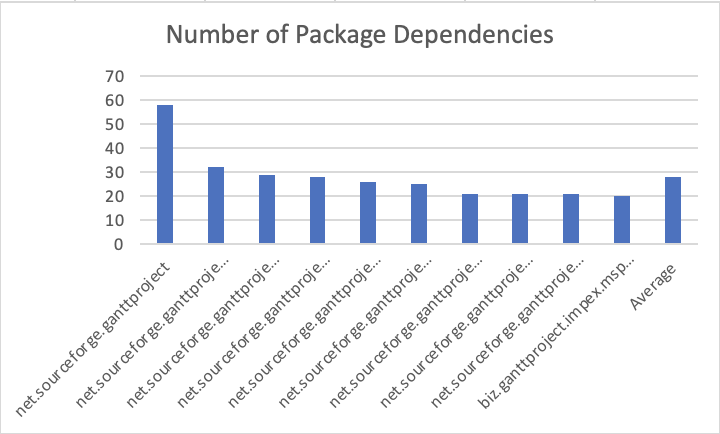
Uma imagem com mesa

Descrição gerada automaticamente



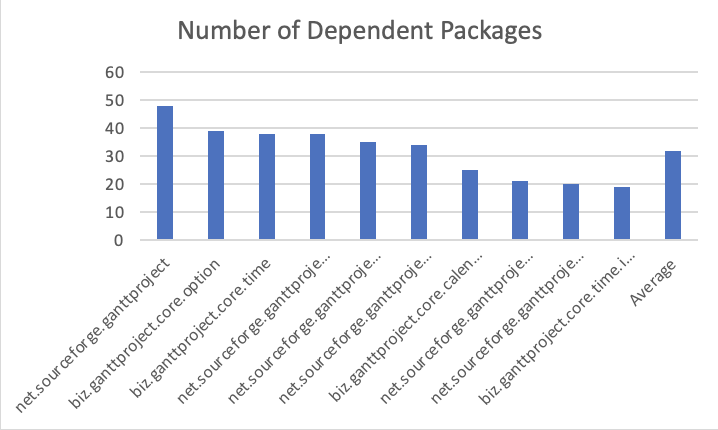
Uma imagem com mesa

Descrição gerada automaticamente



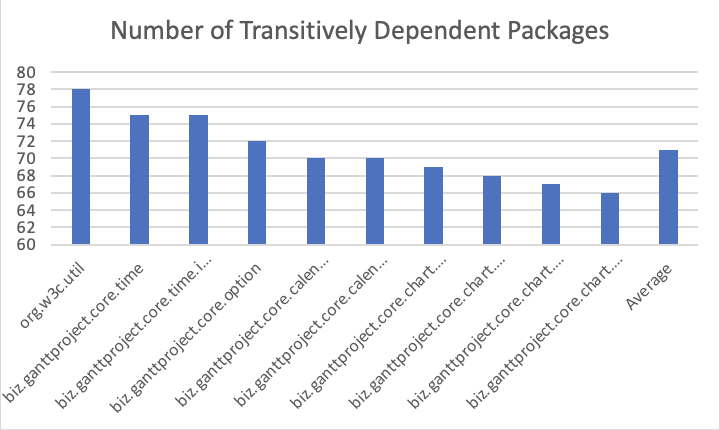
Uma imagem com mesa

Descrição gerada automaticamente



Uma imagem com mesa

Descrição gerada automaticamente



* Analysys:

Having a high number of cyclic dependencies in classes means the class is tightly coupled, going against the design principle of high cohesion and low coupling. If a class has a high number of cyclic dependencies, changing it could generate problems and create a chain reaction across all classes that depend on it.

Having a high number of package dependencies also means that if we change the visibility of one method, variable, or class, other classes might have problems accessing it, making code smells of inappropriate intimacy stand out even more.

Review:

Name and number: Simão Carrasco 59208

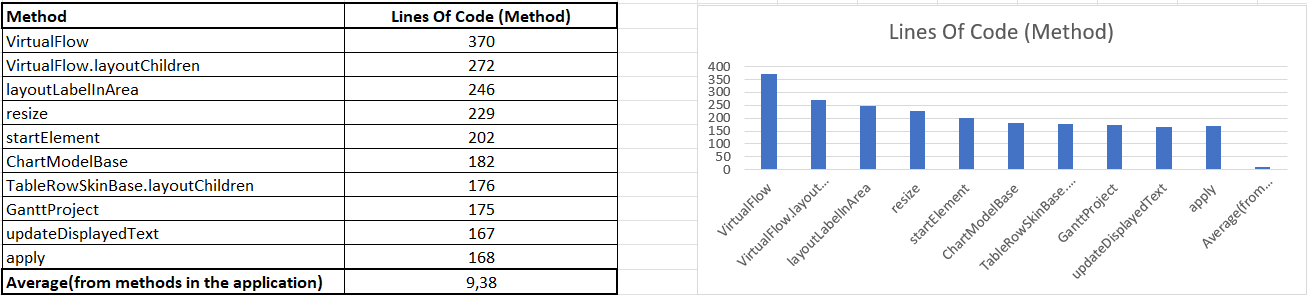
Comprehensive and extensive review.

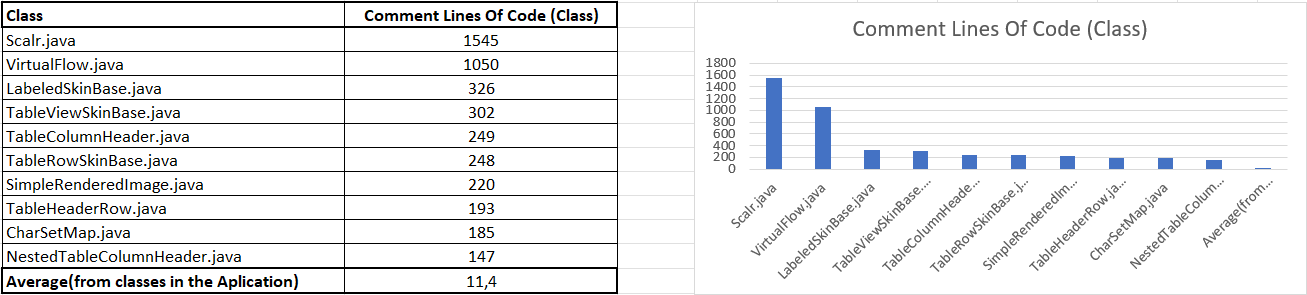
* 58013 – Rodrigo Lopes:
* 59208 – Simão Carrasco:

**Metrics Set: Line Of Code Metrics**

Uma imagem com mesa

Descrição gerada automaticamente





Através desta avaliação, foram descobertas classes e métodos com um número muito elevado de linhas em relação ao resto da aplicação, o que pode significar a presença de Code Smells.

Em classes como Scarl.java, mais de metade das linhas são comentários, o que sugere o Code Smell de demasiados comentários.

O método  layoutLabelInArea, é muito longo (246 linhas), o que sugere o Code Smell de método longo, e além disso a sequência de argumentos x,y,w, e h pode sugerir o Code Smell de data clump.

O Code Smell de método longo já tinha sido identificado numa instância semelhante na primeira fase, em que um método tinha um número elevado de linhas.

**Name and Number:** Inês Carvalho 45345

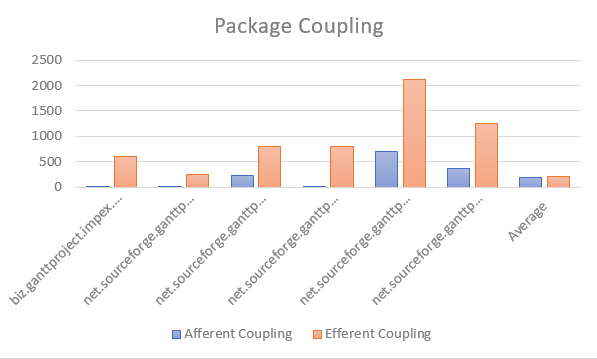
**Review:** Well summarized and concise review correctly identifying the code smells

* 60597 – Pedro Catarino:

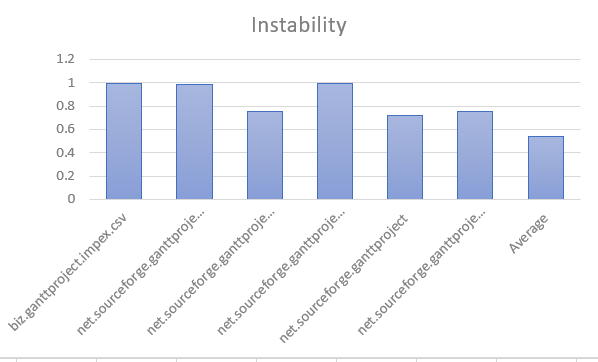
**Metrics Set: Martin Packaging Metrics**

## **Metrics Analysis**

From the data collected, we can underline the packages with big differences between Afferent Coupling, which is the number of classes in other packages that depend upon classes within the package, and Efferent Coupling, which is the number of classes in other packages that the classes in the package depend upon.



Graph – Packages Couplings



Graph 2 – Packages Instability

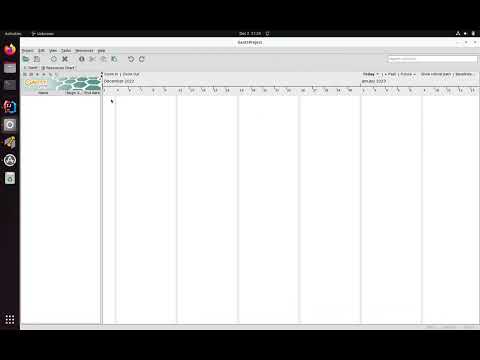
As shown above, we can relate the difference between Efferent Coupling and Afferent Coupling with the instability of the package. Packages with high numbers of Efferent Coupling and low numbers of Afferent Coupling have higher instability, since they depend a lot on classes in other packages but do not have many classes from different classes depend on it.

## **Metrics Conclusions**

From the metrics results, we can conclude that the package dependencies might not be well distributed, given that some packages are highly dependent on others, making them unstable. When a package is highly dependent on other, changes on the “other” package will affect it, and may result in code smells and other issues.

**Functionality 1:** An option that shows the number of tasks of the current day.

**Functionality 2:** While opening a project, emit a warning if there is at least one task with less than 80% completion percentage and it’s end date is the day after or before.

**Demo Video**[](https://www.youtube.com/embed/Wa-OSvNXP2s?feature=oembed)