

DESIGN PATTERNS EARTH

MARCO SANTIAGO CANCHÉ MAY

JOSÉ LUIS GUTIÉRREZ COUOH

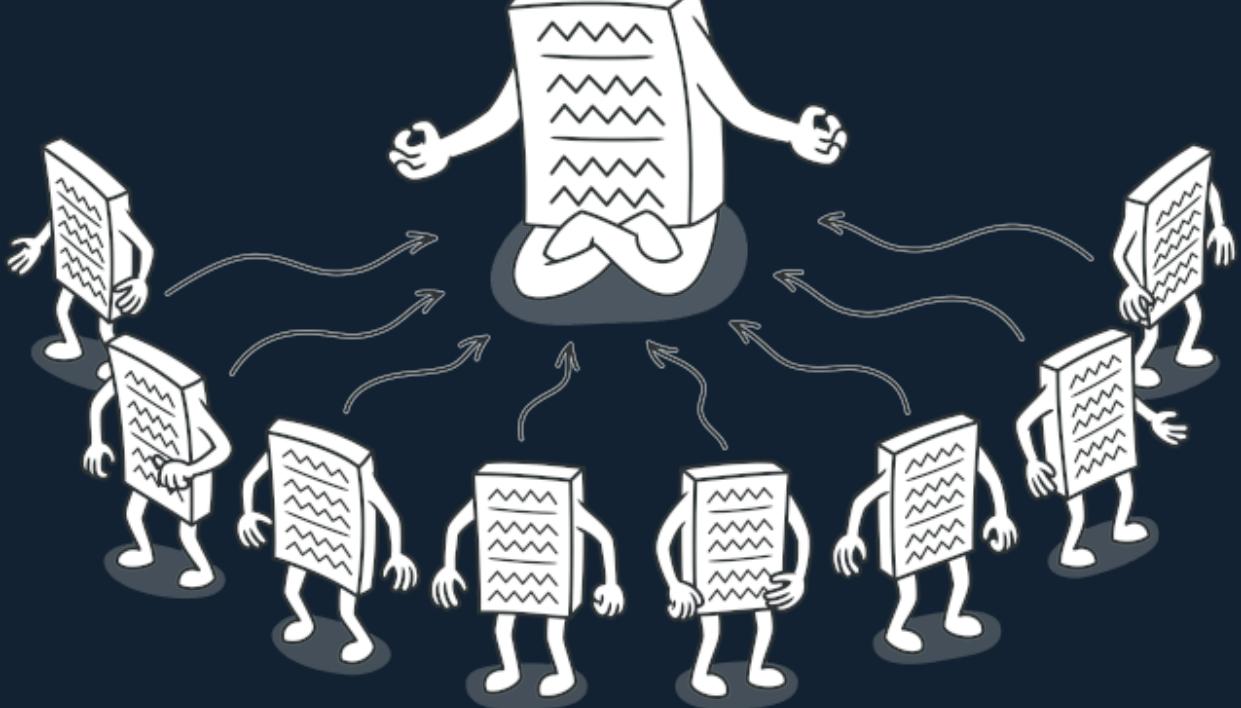
RODRIGO JOAQUÍN PACAB CANÚL

ORLANDO ISAÍAS RODRÍGUEZ COUOH

CARLOS ISREAL RUZ RUZ



SOFTWARE DESIGN PATTERNS



GOOD INFORMATION WITHOUT REAL EXAMPLES





REFACTORING · GURU ·

★ Contenido premium

💡 Patrones de diseño

Introducción

Catálogo

Patrones creacionales

Patrones estructurales

Patrones de comportamiento

Ejemplos de código

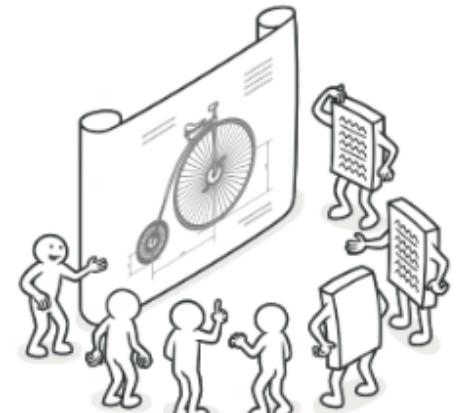
⚡ Refactorización
(próximamente)

👤 Iniciar sesión

✉️ Contáctanos



PATRONES de DISEÑO



Los **patrones de diseño** (design patterns) son soluciones habituales a problemas comunes en el diseño de software. Cada patrón es como un plano que se puede personalizar para resolver un problema de diseño particular de tu código.

[¿Qué es un patrón de diseño?](#)

Ventajas de los patrones

Los patrones son un juego de herramientas que brindan soluciones a problemas habituales en el diseño de software. Definen un lenguaje común que ayuda a tu equipo a comunicarse con más eficiencia.

[Más sobre las ventajas »](#)

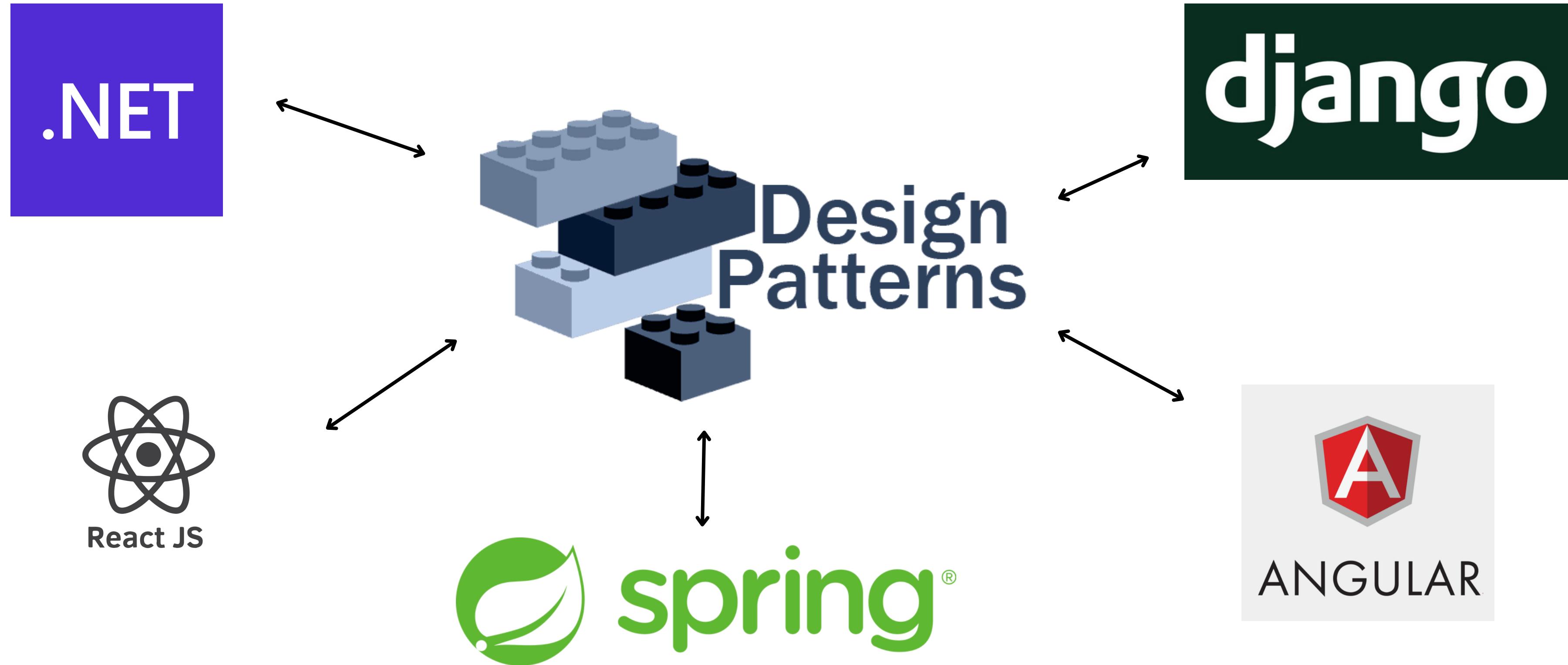
Catálogo de patrones

Lista de 22 patrones de diseño clásicos, agrupados con base en su propósito.

[Consulta el catálogo »](#)

Más sobre categorías »

OUR IDEA



SYSTEM OBJECTIVES



Clear and Concrete
Resources

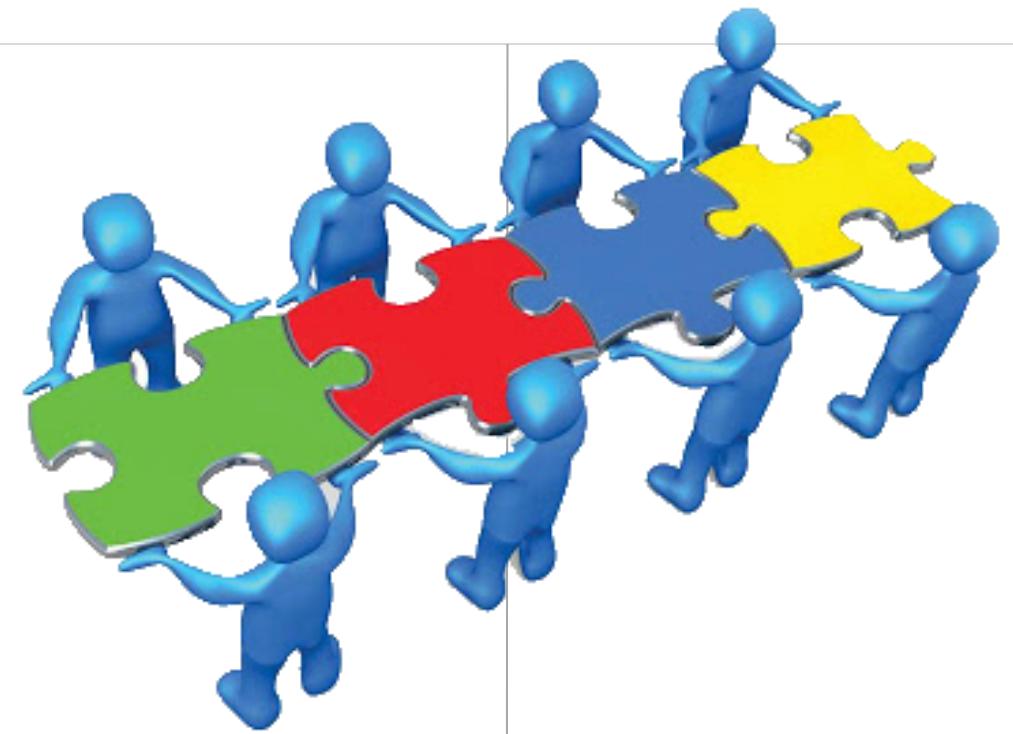
A screenshot of a code editor window. On the left, the file structure shows a folder named 'INDUSTRIAL' containing several CSS files like 'estilos.css', 'fullpage.css', and 'index.html'. The main area displays two files: 'estilos.css' and 'index.html'. The 'estilos.css' file contains CSS rules for various elements such as fonts, colors, and layout. The 'index.html' file contains the HTML structure of a page, including a header, a section with a background image, and a footer. The code editor has tabs for 'Archivos', 'Editor', 'Selección', 'Ver', 'Depurar', 'Terminal', and 'Ayuda' at the top.

Patterns with Real-Life
Implementations

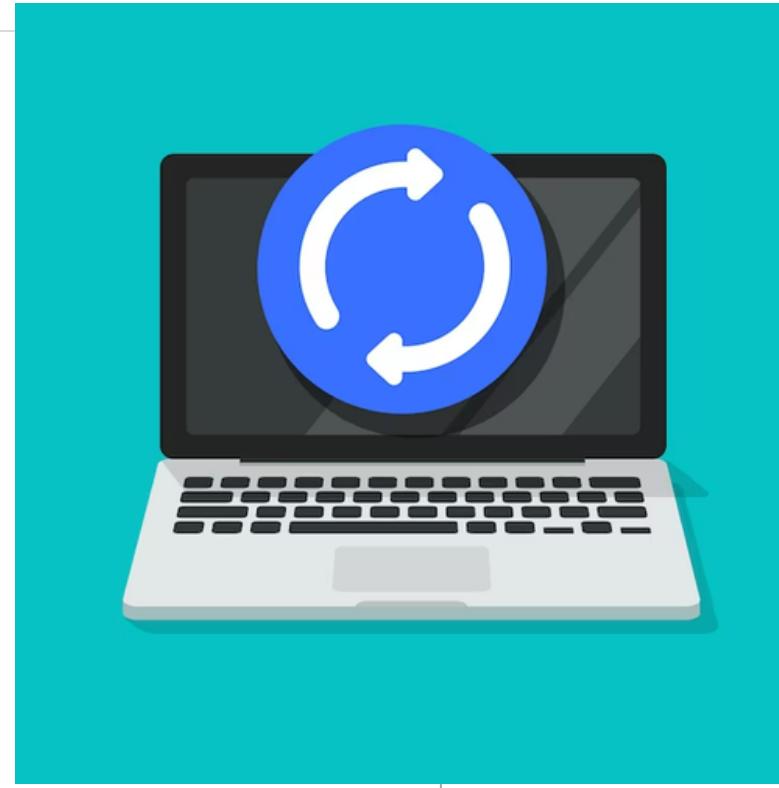
SOLID
Design Principles

Identify the SOLID
principles that each
pattern uses.

SYSTEM LIMITATIONS



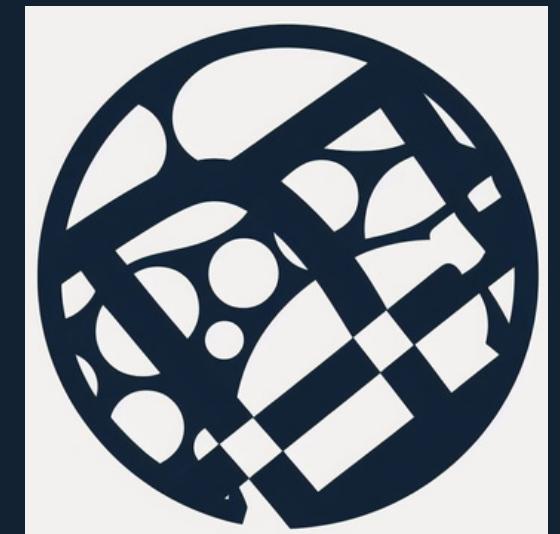
**Does not allow
collaboration**



**No updates in a
row**

DESIGN PATTERNS EARTH

HOW WAS THE PROCESS?



3 WELL-DEFINED PHASES

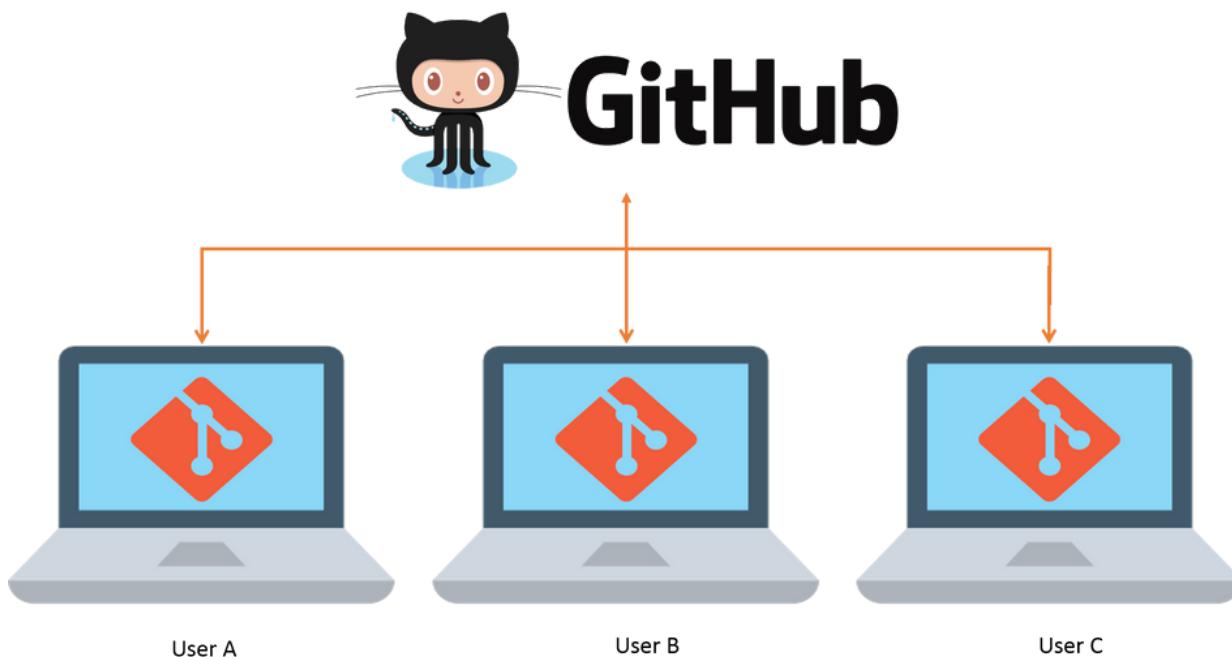
- 1 IDEAS PHASE - REPOSITORY STRUCTURE
- 2 RESEARCH PHASE
- 3 ARTIFACTS PHASE

1- IDEAS PHASE

REPOSITORY STRUCTURE



Approval for the
project idea



Protected branches

A screenshot of a GitHub repository page for 'hafeezulkareem / delete_demo'. The page shows a single file, 'README.md', which contains the text 'A repo to demonstrate the deletion.' The repository has 1 commit and 1 branch. The interface includes standard GitHub navigation like 'Code', 'Issues', and 'Marketplace'.

Set up the repository structure

A screenshot of the 'Conventional Commits' extension page in the Visual Studio Marketplace. It shows the extension version v1.25.0, developer vivaxy, and a rating of 5 stars from 25 reviews. The page includes sections for 'VSCode Conventional Commits' and 'Features', listing support for commitlint configs, auto commit and push, project level scope management, gitmojis, and VSCode workspaces.

Good practices with
commits

2- ·RESEARCH PHASE

AND TEAM ORGANITATION

RODRIGO PACAB
SCRUM MASTER

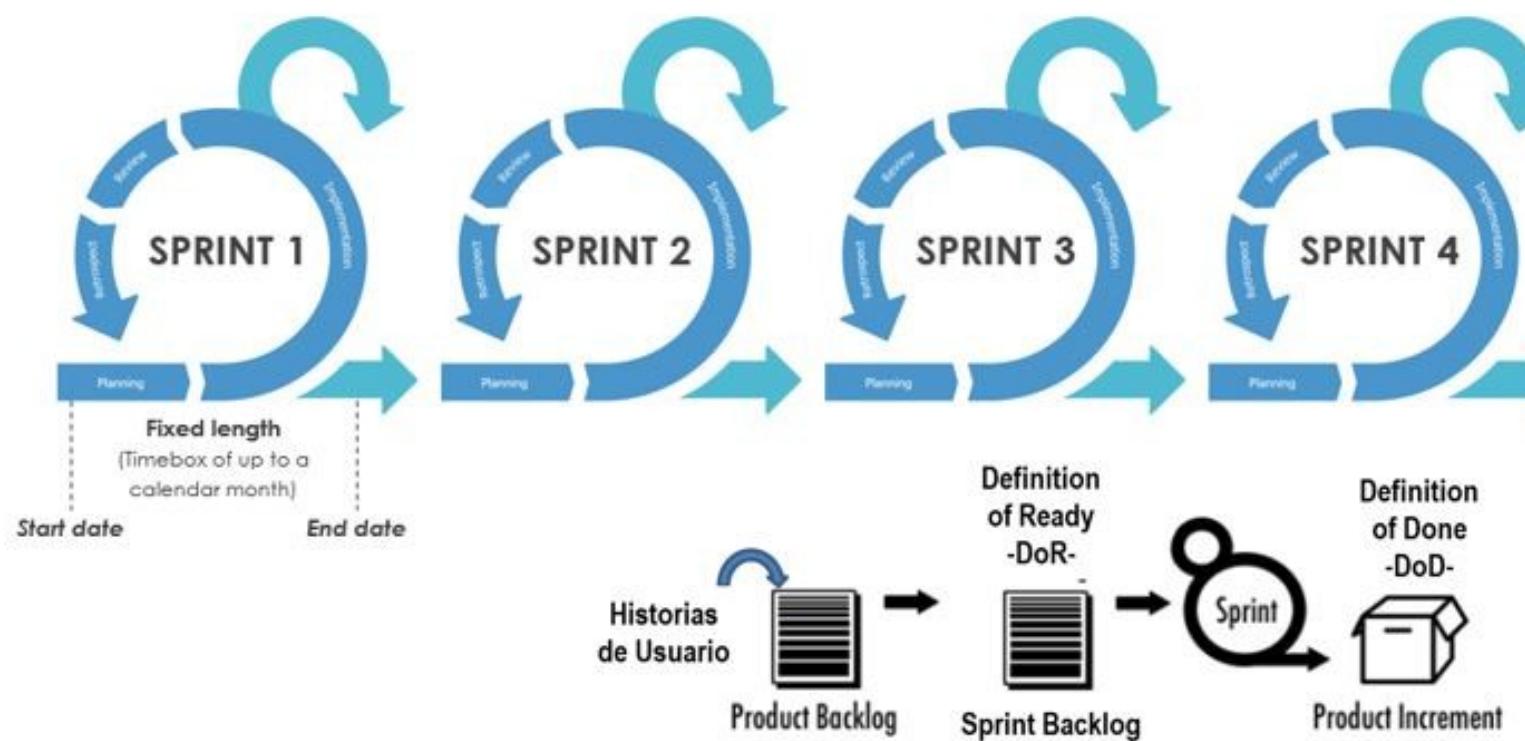
ISAIAS RODRIGUEZ
PRODUCT OWNER

MARCO CANCHÉ
SCRUM MASTER - PRODUCT OWNER

CARLOS RUZ
DEVELOPER

JOSÉ LUIS GUTIÉRREZ
DEVELOPER

SCRUM



Screenshot of a digital tool interface showing a sprint backlog for the project "OctoArcade Invaders".

The interface includes:

- Header:** Planning, Sprint Board, Roadmap, Alpha, My work, Features, Priority, By person, Status Board, By status, By Sprint, Date fields, Month, Today.
- Calendar:** January 2023, showing dates from 11 to 31.
- Sprint Details:** Sprint 1, "Account subscription des... #818 (1 of 3)" (Tue, Jan 24 - Sun, Feb 5).
- Tasks:** Final UX tweaks #1247, Documentation and Support #831.
- Sprint 2:** "Beta Launch #1240 (1 of 5)" (Thu, Jan 19 - Tue, Feb 7). Tasks include: Beta signup page #824, General bug fixes from Alpha feedback #808, Tweak to difficulty level based on Alpha #826, Updates and bug fixes to engine from Beta #823, Beta go-no-go meeting #810.
- Sprint 3:** "Updates and bug fixes to engine from Beta #823" (Sat, Dec 3 - Sun, Jan 29, 2023).

3- ARTIFACTS PHASE

Requirements

Functional requirements

1. **Design pattern in software:** The software must implement a specific set of design patterns, including but not limited to, Singleton, Factory Method, Observer, Strategy, Proxy, Memento, Command, Mock Object, Repository (or DAO), among others, as appropriate.
2. **Implementation Examples:** In each design pattern example, the software code must be shown with specific examples (observations and comments) that show the application of said pattern in situations relevant to the project.
3. **Interaction between Patterns:** If possible, it should be demonstrated how the different design patterns interact with each other in the software to achieve a coherent and efficient design.
4. **Detailed Documentation:** Each design pattern shown must be accompanied by detailed technical documentation that explains its use, advantages, disadvantages and how it is integrated into the overall project.
5. **Compliance with Coding Standards:** The code that implements the design patterns must follow the coding standards (ours) defined for the acceptance of the project code.

Non-functional requirements

1. **Usability:** The system must work on different platforms
2. **Availability:** The software must be available 24 hours a day, 7 days a week.

We obtained these requirements

ARTIFACTS

CLASS DIAGRAM

First Iteration of classes

As this is the first version of the classes, they are currently few and somewhat general. With each iteration, they will be further developed and become more complex in their relationships.

1. DesignPattern Class:

Attributes:

- name (Type: String): Stores the name of the design pattern.
- description (Type: String): Holds a detailed description of the pattern.
- patternType (Type: String): Indicates the category to which the pattern belongs.
- purpose (Type: String): Describes the purpose of the design pattern.
- CodeExample (Type: Code Example): Possibly a reference to an object of the [CodeExample](#) class related to the pattern.

Methods:

- Constructor to initialize the attributes.
- Methods to get and set each attribute.
- Method to display the complete information of the pattern.

2. CodeExample Class:

Attributes:

- title (Type: String): A descriptive title for the example.
- description (Type: String): A brief description of the example.
- code (Type: String): The example code related to the pattern.

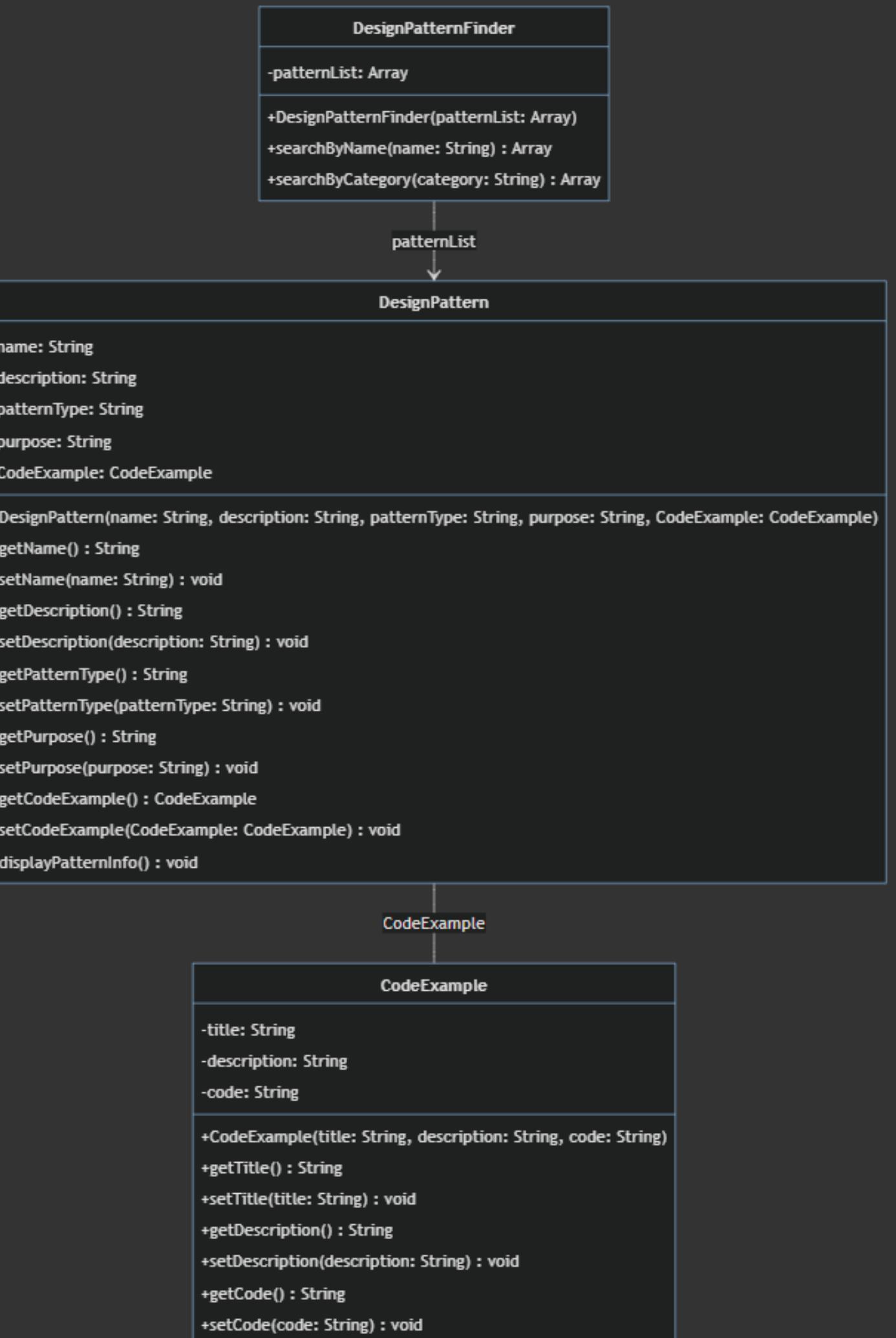
3. DesignPatternFinder Class:

Attributes:

- patternList (Type: Array of references to [DesignPattern](#)): Stores a list of objects related to design patterns.

Methods:

- searchByName(name: String): List of [DesignPattern](#): Allows searching for design patterns by name.
- searchByCategory(category: String): List of [DesignPattern](#): Facilitates the search for design patterns by category (e.g., creational, structural, behavioral, etc.).



ARTIFACTS

USER STORIES

User stories

User Story 1 - Implementation of Design Patterns

As a project developer, I want the software to implement a variety of design patterns, including but not limited to Singleton, Factory Method, Observer, Strategy, Proxy, Memento, Command, Mock Object, Repository (or DAO), as appropriate.

User Story 2 - Interaction Between Design Patterns

As a project developer, I want to demonstrate how different design patterns interact with each other in the software to achieve a coherent and efficient design.

User Story 3 - Detailed Documentation

As a project developer, I want to create detailed technical documentation for each design pattern implemented in the software. This documentation should explain its usage, advantages, disadvantages, and how it integrates into the project overall.

User Story 4 - Compliance with Coding Standards

As a project developer, I want to ensure that the code implementing the design patterns complies with the coding standards defined for the acceptance of the project.

User Story 5 - Usability on Different Platforms

As a general user, I want the software to be usable on different platforms to ensure accessibility and convenience.

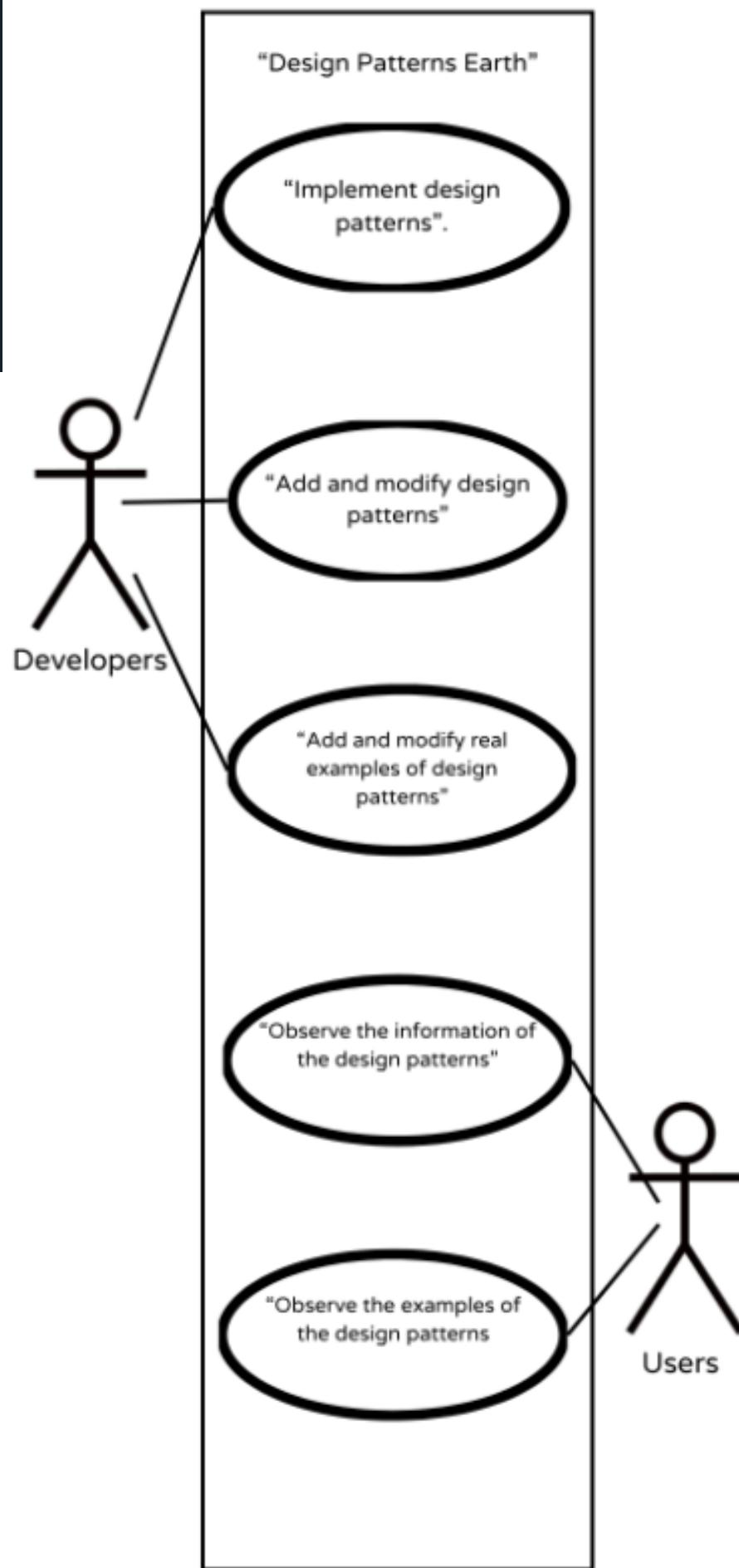
User Story 6 - 24/7 Availability

As a general user, I want the software to be available 24 hours a day, 7 days a week, to ensure continuous availability.

ARTIFACTS

USES CASES DIAGRAM

Use cases diagram

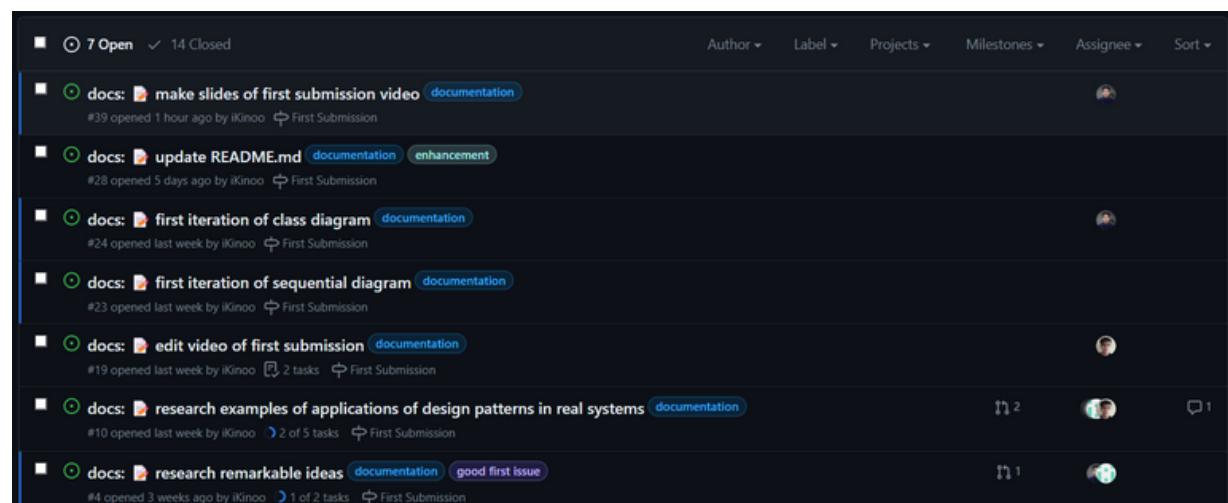
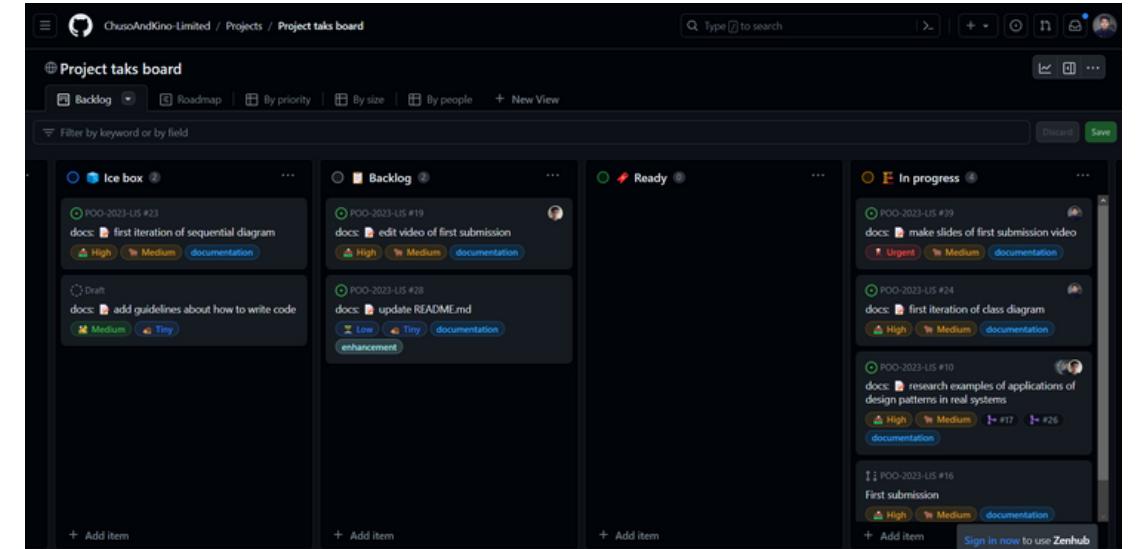
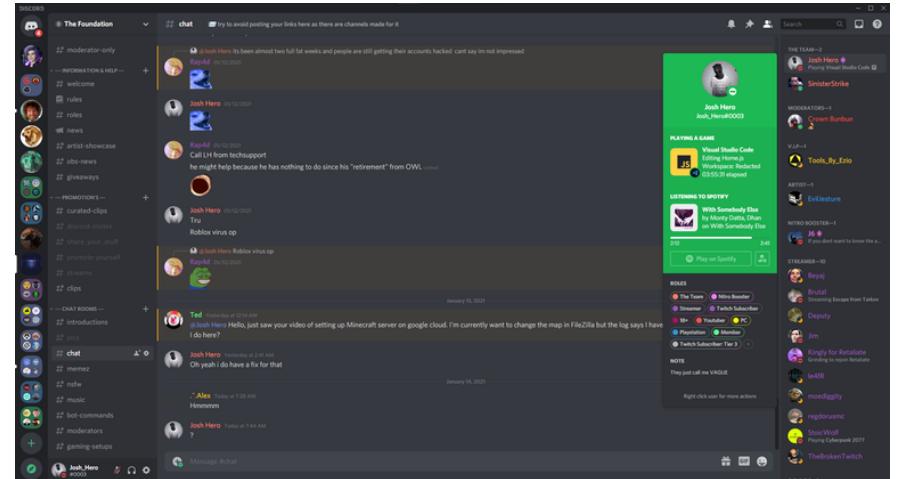
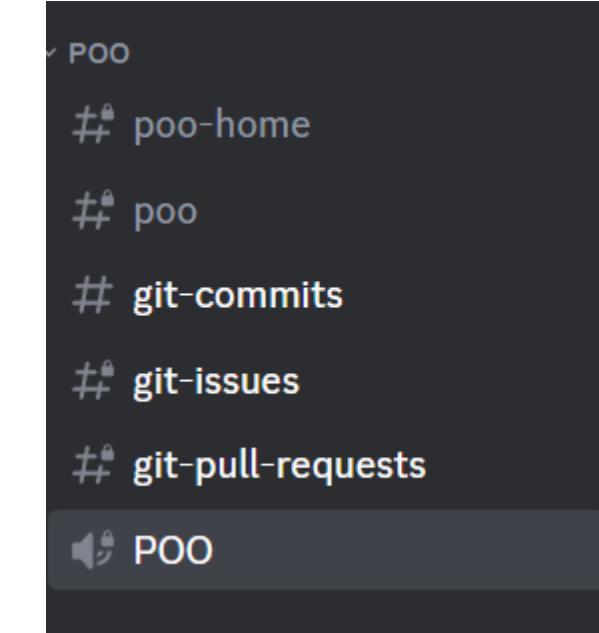
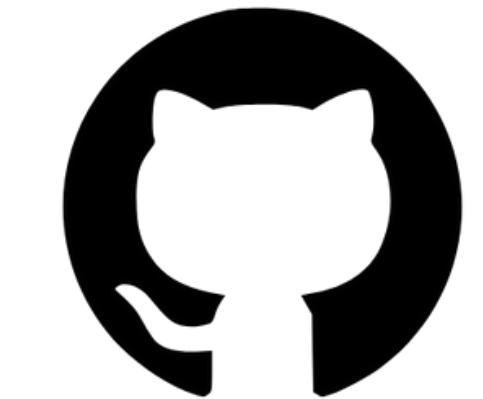


PROCESS MANAGEMENT

RESULTADOS

¿CÓMO PODEMOS MEDIR
NUESTRO IMPACTO?

Identificamos nuestros métricas principales y
objetivos de negocio



WE NEEDED AN ORGANIZATION



Chuso & Kino Limited SA de CV

More than one person has privileges

DISCORD



A screenshot of a Discord channel titled '#git-commits'. It shows several messages from GitHub's bot (@GitHub) and other users (IsaiasRdzc, iKinoo, Josegutierrezcouch). The messages are related to Git commits, pull requests, and documentation updates for a project named 'POO-2023-LIS'.

For commits

A screenshot of a Discord channel titled '#git-issues'. It shows messages from GitHub's bot (@GitHub) and user MarcoSIIIU. The messages discuss new comments on issues and pull requests, specifically regarding process descriptions and management.

For issues

A screenshot of a Discord channel titled '#git-pull-requests'. It shows messages from GitHub's bot (@GitHub) and user IsaiasRdzc. The messages track the status of a pull request (#36), including its opening, review, and closure.

For pull request

GITHUB PROJECTS

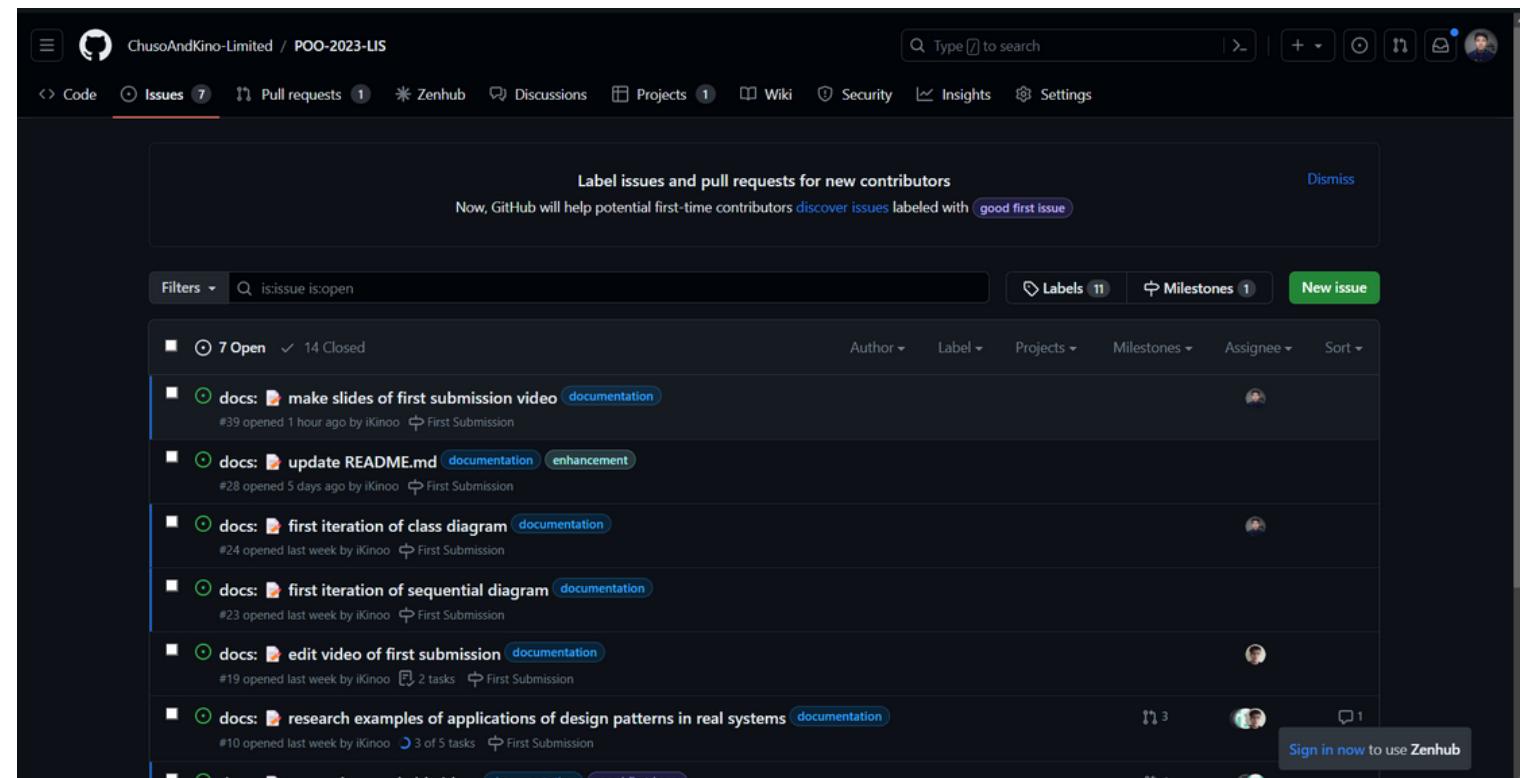
The screenshot shows the GitHub Project tasks board interface. At the top, there are navigation links for Backlog, Roadmap, By priority, By size, By people, and New View. A search bar and filter options are also present. Below the header, there are four columns representing different stages: Ice box, Backlog, Ready, and In progress. Each column contains several items, each with a title, description, and status indicators (e.g., High, Medium, documentation). Buttons for adding new items are located at the bottom of each stage column.

Backlog with Kanban

The screenshot shows the GitHub Project tasks board interface with a Roadmap view. The top navigation bar includes Backlog, Roadmap, By priority, By size, By people, and New View. A search bar and filter options are also present. The main area displays a calendar grid for September and October 2023. Each day in the grid has a list of tasks assigned to it. The tasks are color-coded by priority (green for High, orange for Medium, blue for Low) and include descriptions like "docs: first iteration of sequential diagram" and "docs: edit video of first submission". A "First Submission" label is placed over specific tasks on October 30th. Buttons for adding new items are located at the bottom of the grid.

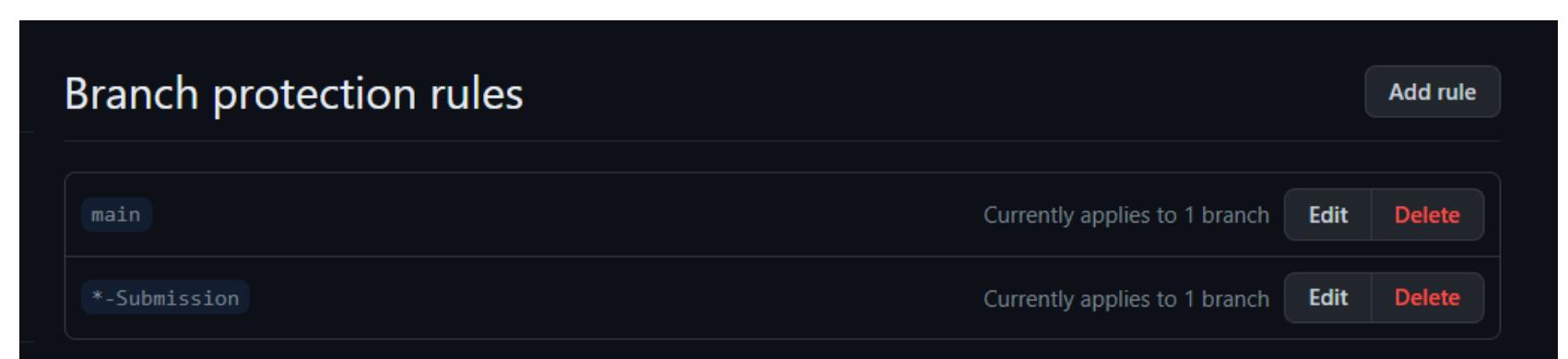
Roadmap view (sprints)

GITHUB PROJECTS



A screenshot of the GitHub Issues page for the repository 'ChusoAndKino-Limited / POO-2023-US'. The page shows a list of 7 open issues. A prominent modal at the top left says 'Label issues and pull requests for new contributors' and 'Now, GitHub will help potential first-time contributors discover issues labeled with good first issue'. The issues listed are all labeled 'docs' and include titles like 'make slides of first submission video', 'update README.md', 'first iteration of class diagram', 'first iteration of sequential diagram', 'edit video of first submission', and 'research examples of applications of design patterns in real systems'. Each issue has a small thumbnail, the title, a link to the pull request, and a timestamp.

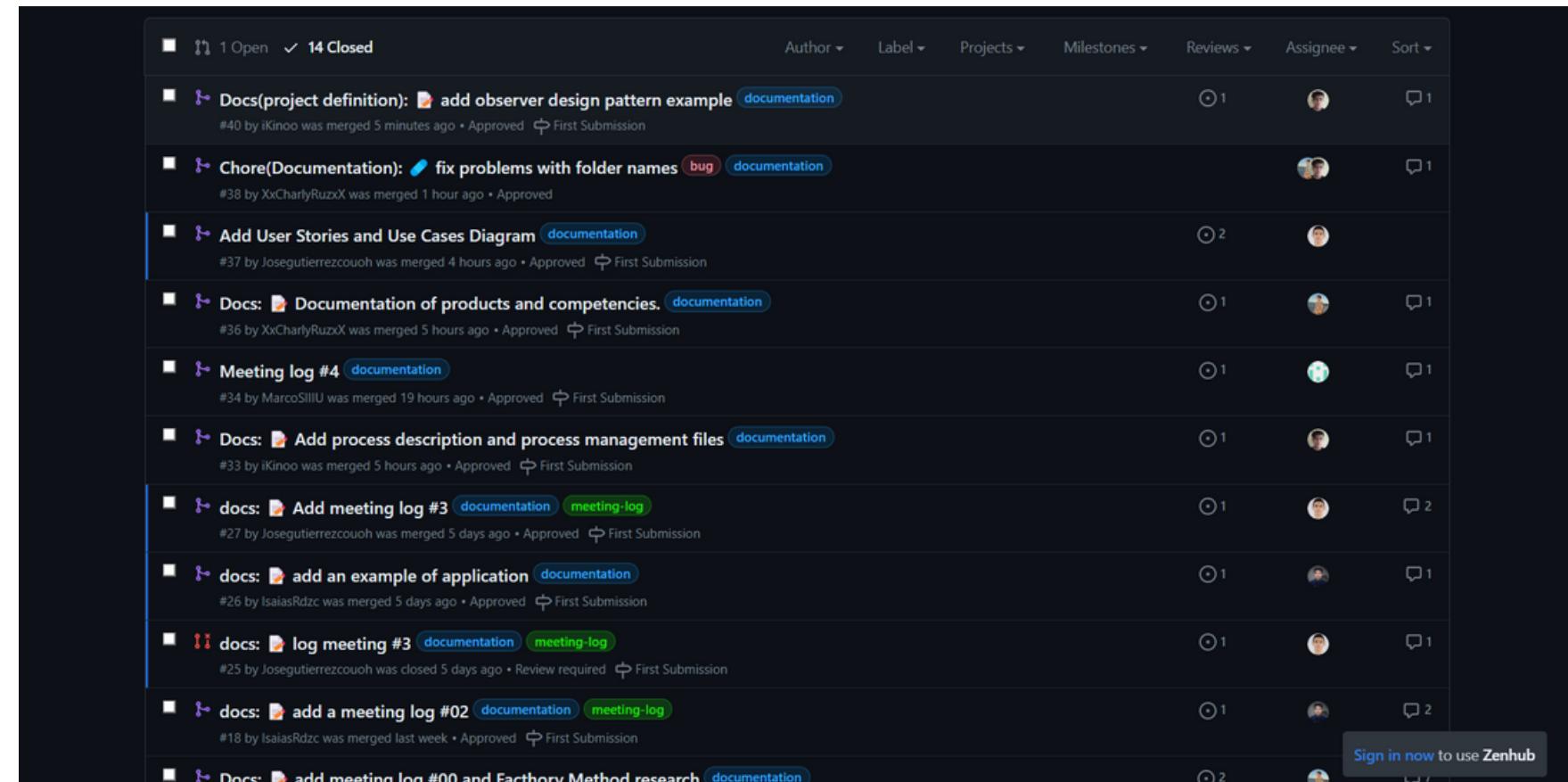
Issue was manipulated as one task or
a group of tasks



A screenshot of the GitHub Branch protection rules page. It shows two rules: one for the 'main' branch and one for '* -Submission'. Both rules apply to 1 branch. Each rule has an 'Edit' button and a 'Delete' button. The 'main' rule is currently applied to the 'main' branch, and the '* -Submission' rule is currently applied to '* -Submission'.

Protected branches

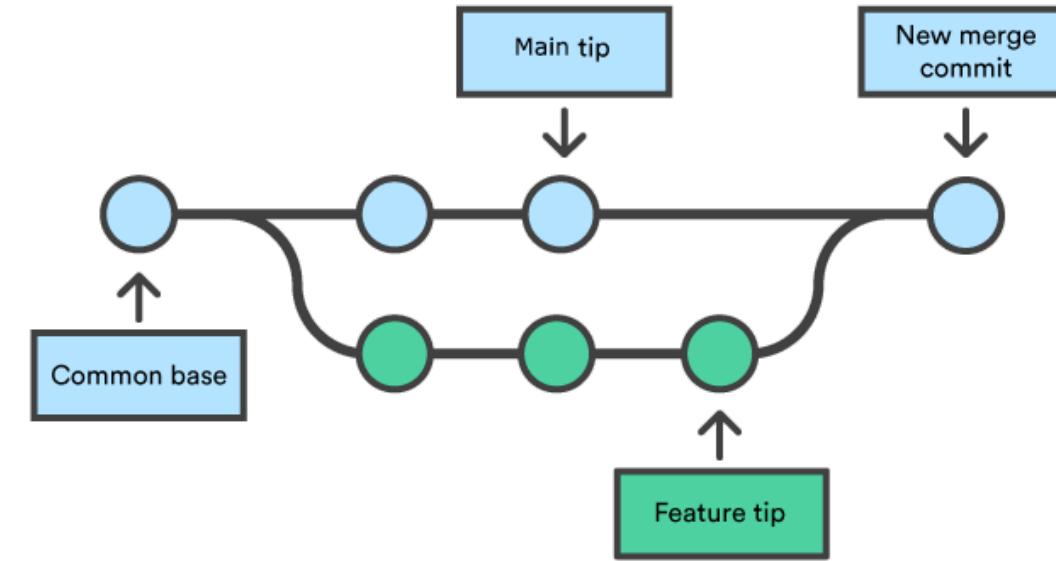
GITHUB PROJECTS



A screenshot of a GitHub Project board. The top navigation bar shows '1 Open' and '14 Closed'. The board lists several pull requests:

- #40 by iKinoo was merged 5 minutes ago • Approved → First Submission
- #38 by XxCharlyRuzX was merged 1 hour ago • Approved
- #37 by Josegutierrezcouh was merged 4 hours ago • Approved → First Submission
- #36 by XxCharlyRuzX was merged 5 hours ago • Approved → First Submission
- #34 by MarcoSILIU was merged 19 hours ago • Approved → First Submission
- #33 by iKinoo was merged 5 hours ago • Approved → First Submission
- #27 by Josegutierrezcouh was merged 5 days ago • Approved → First Submission
- #26 by IsaliasRdzc was merged 5 days ago • Approved → First Submission
- #25 by Josegutierrezcouh was closed 5 days ago • Review required → First Submission
- #18 by IsaliasRdzc was merged last week • Approved → First Submission
- #00 and Factory Method research documentation

At the bottom right, there is a 'Sign in now to use Zenhub' button.



A contributor could only merge their work into the first-submission branch through a Pull Request, and this Pull Request must be approved at least by one person