Object-Oriented Programming Project

Angie Nathaly Cisneros Guerrero	20242020084
Luis Sebastián Correa Barreto	20242020085

Project: Replicate the functionality of the "El Espectador" website, which provides access to current news.

1. **Conceptual Design Updates:**

- **Objetives:**
    - Keep the user informed through easily accessible news.
    - Find news by classification to make your search easier.
    - Offer comprehensive and detailed reports on relevant topics.

- **Functional requirements:**
    - View the main news on the homepage.
    - View top news.
    - See the different news sections.
    - View latest news (last 9 published).
    - Access full details of a news article.

- **Non-Functional requirements:**
    - Scalability for increased news content.
    - Intuitive and accessible navigation (for demographics).
    - Modular and maintainable codebase.
    - Adequate documentation of code and architecture.

- **User stories:**

Title: Home Page Reader	**Priority:** High	**Estimate:** 6 weeks
User Story: ***As a visitor***, I want to see the main and top news clearly on the homepage so I can quickly get informed about the most relevant events.		
Acceptance Criteria: -Main news article displayed prominently. - Section for "Top News" visible under the main article.		
Title: Latest News Reader	**Priority:** Medium	**Estimate:** 4 weeks
User Story: ***-As a*** frequent visitor, I want to view the latest news quickly to stay updated.		
Acceptance Criteria: - “Latest News” section that dislays the 9 most recently published articles. - Articles ordered form newest to oldest.		

Title: International News	**Priority:** Low	**Estimate:** 4 weeks
User Story: ***As an*** international affairs enthusiast, I want a dedicated section for international news so I can stay informed globally.		
Acceptance Criteria: - International" section easily accessible. - Displays at least 9 articles from different countries.		

Title: News	**Priority:** High	**Estimate:** 1 week
User Story: ***As an*** app developer I want to create news whit title, content and date so the readers can see the info sorted and updated.		
Acceptance Criteria: - “News” this is the most important thing in the project because it’s the base for all sections.		

Title: Different types of sections	**Priority:** Medium	**Estimate:** 4 weeks
User Story: ***-As a*** frequent visitor, I want to view a section of latest news and another I don't mind whatever it is.		
Acceptance Criteria: -This is acceptable and good because we can create a single class called "Section" to spare different kind of news		

- **CRC Cards:**

Class Name: New	
Responsibilities:	Collaborators:
-Manage article attributes (Title, content, date, etc.)	-Section.

Class Name: Section	
Responsibilities:	Collaborators:
-Group news by category (Latest news, international news).	-News.

Class Name: Homepage	
Responsibilities:	Collaborators:
-Display last news	-News -Section

Class Name: LatestNewsSection	
Responsibilities:	Collaborators:
-Group news by category (Latest news, international news).	-News. -Section

- **Documentation of changes respect to workshop-1:**

❖ Decisions emerged from further planning and discussions.

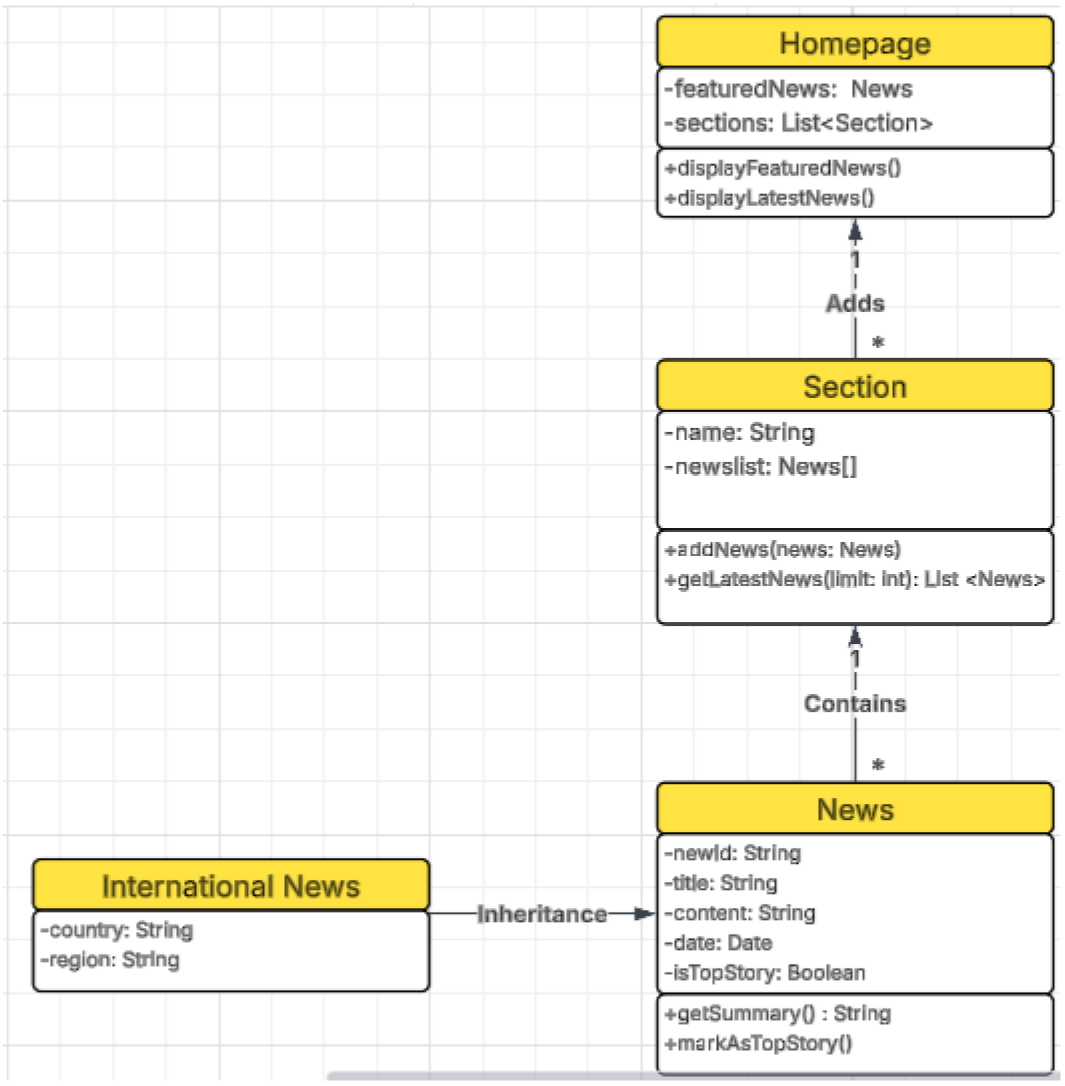
Workshop 1 → Workshop 2: Evolution of UML & OOP Implementation:

-Objective: Translate the initial conceptual design into a technical blueprint using OOP principles.

- Refined Class Diagram
- Added OOP Pillars
- New Components
- User Story Alignment
- CRC card classes (Section, News, Homepage, International news).
- Enhanced responsibilities

2. **Technical design:**

- **UML diagram; there's hierarchies, class members (attributes, methods), and relationships.**



- **How classes implement or override methods to fulfill user requirements and how they integrate the OOP principles:**

News Class:

User Requirements Fulfilled:

- "See news articles" (via getSummary() method).

OOP Principles Applied:

- Encapsulation: Private attributes with private getters/setters.
- Reusability: Base class for all news types.

Key Methods:

- getSummary(): Returns title + date (core news display).
- markAsTopStory(): Flags important news (for homepage).

InternationalNews Class (extends News):

User Requirements:

- "International news section with country tag".

OOP Principles:

- Inheritance: Extends News class.
- Polymorphism: Overrides getSummary() to include location.

Key Methods:

- getSummary(): "[Country] Title (Date)" format.
- (Inherits all News methods).

Section Class:

User Requirements:

- "Group news by category".
- "Latest news section".

OOP Principles:

- Composition: Manages collection of News objects.
- Single Responsibility: Handles news categorization only.

Key Methods:

- getLatestNews(limit): Returns N most recent news.
- getNewsByDate(): Filters by date.
- addNews(): Adds to category.

Homepage Class:

User Requirements:

- "Homepage with featured news".
- "Display latest news".

OOP Principles:

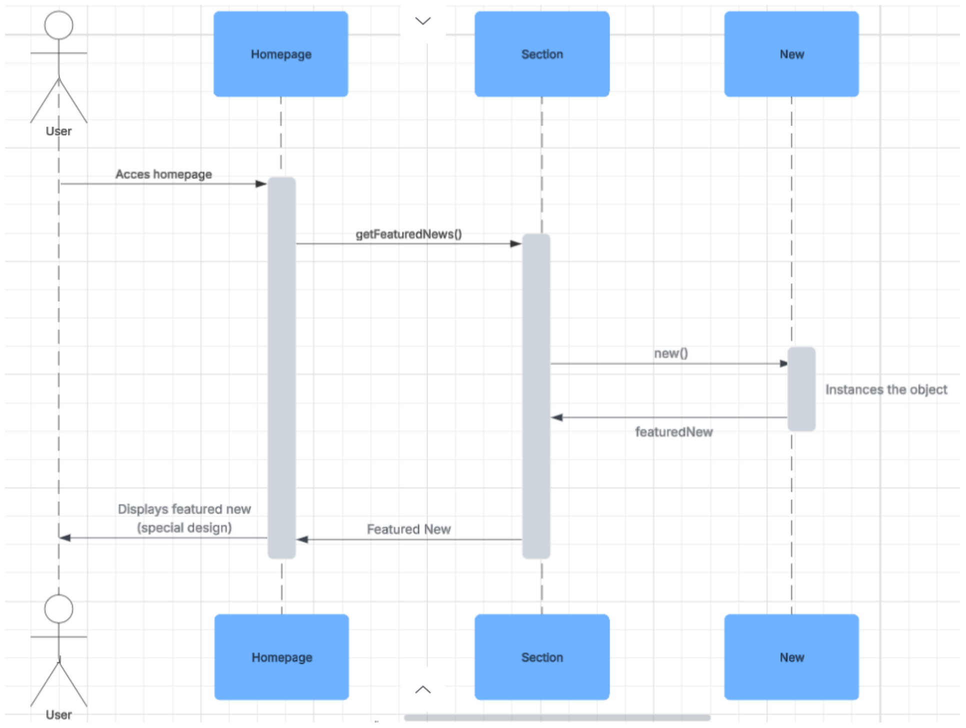
- Dependency Injection: Depends on News/Section.
- Delegation: Uses Section's methods for lists.

Key Methods:

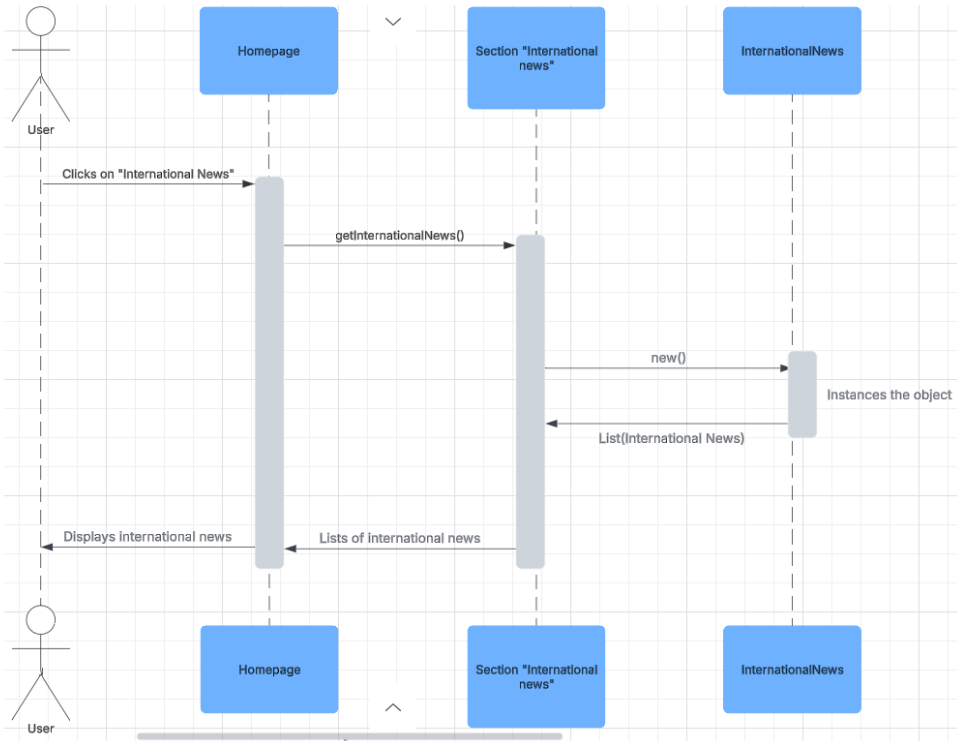
- displayFeaturedNews(): Shows top story.
- displayLatestNews(): Shows recent news (delegates to Section).

This implementation satisfies all specified user stories while maintaining proper OOP design. Each class has clear responsibilities and collaborates through well-defined interfaces.

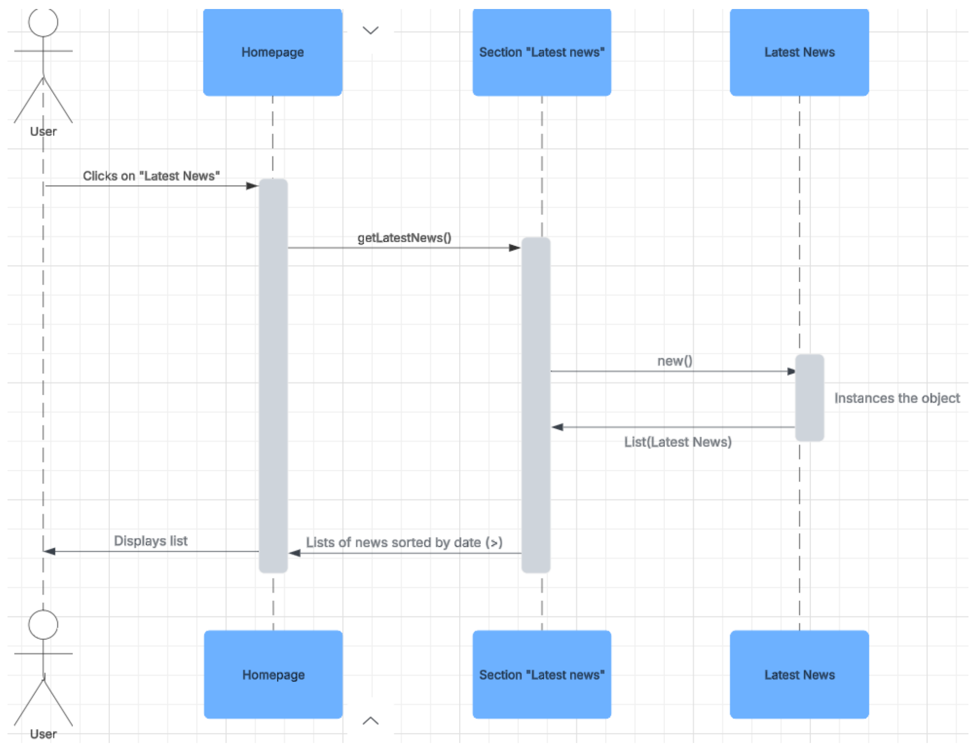
- **Sequences diagrams; these illustrate interactions between classes and the flow of data:**
  - **Diagram “Homepage”:**



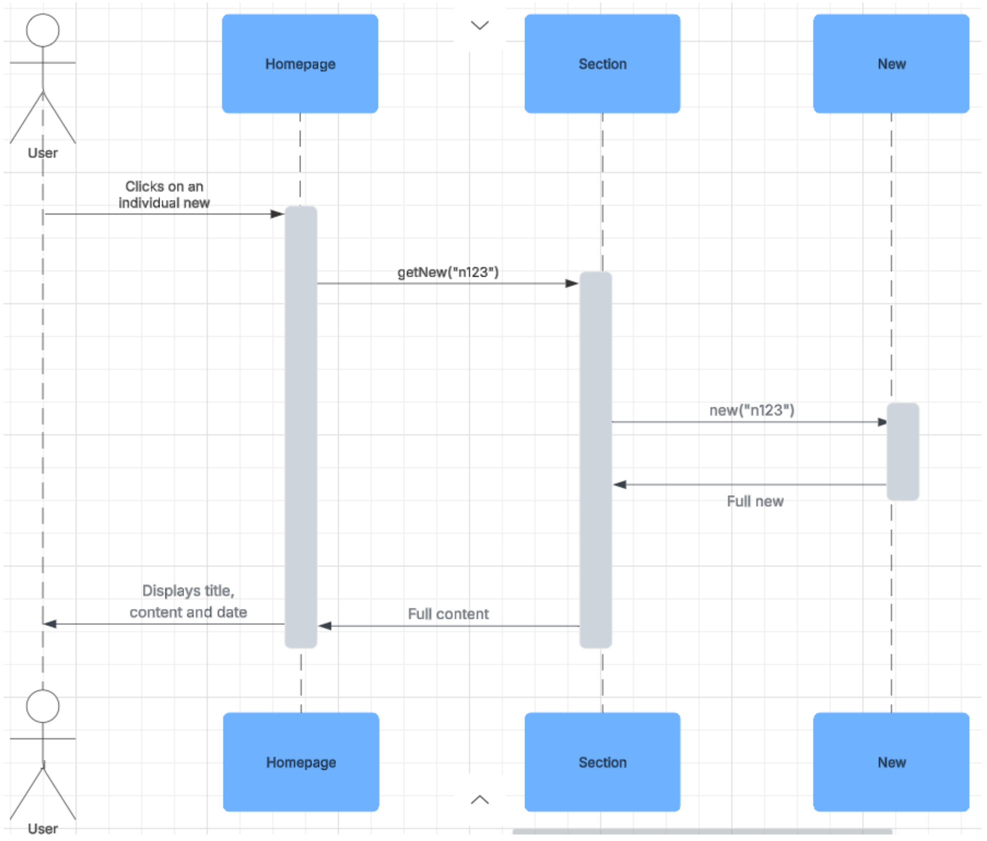
  - **Diagram “Access to international news”:**


User
Homepage
Section "International news"
InternationalNews
Clicks on "International News"
getInternationalNews()
new()
Instances the object
List(International News)
Displays international news
Lists of international news
User
Homepage
Section "International news"
InternationalNews

- **Diagram “Access to latest news”:**



- **Diagram display individual new:**



3. **Implementation Plan for OOP Concepts:**

Summary of how our code will realize encapsulation, inheritance and polymorphism:

- **Encapsulation:** Protecting Data
  - Each class controls access to its internal attributes through private methods. For example, the News class keeps its title and content as private properties, only allowing modifications through methods that include validations (like ensuring the title isn't empty).
- **Inheritance:** Specializing Behavior
  - The InternationalNews class inherits from the News, leveraging all its base attributes and methods while adding specific properties like country of origin, demonstrating the principle of extending without modifying.
- **Polymorphism:** Flexible Behavior
  - When the system calls the "getSummary" method, the InternationalNews class responds differently than the base News class by automatically including country information. This allows processing different news types uniformly.
- **Composition:** Building Relationships
  - The Section class contains and manages collections of News objects, coordinating how news is grouped and retrieved. The Homepage in turn uses these sections to present information to users.
- **System Flow:**
  - Creation: News instances are created with their basic data.
  - Organization: News is assigned to thematic sections.
  - Presentation: The Homepage requests news from sections and displays it.
  - Interaction: Users can navigate through the app without any kind of log in.
- **Advantages of This Design:**
  - Simplicity: Working entirely in-memory avoids infrastructure complexity.
  - Flexibility: Easy to extend with new news types.
  - Maintainability: Each class has clear, well-defined responsibilities.
  - Scalability: The structure easily accommodates new features.

- **Logical Structure:**
  - Models: News (base), InternationalNews (specialized).
  - Managers: Section (organizes news).
  - Interface: Homepage (presents information).
- This approach leverages OOP principles to create a cohesive system where components collaborate while maintaining independence, ideal for projects requiring clarity and evolution capability. The complete in-memory implementation ensures lightweight operation.

-Preliminary directory structure:

Bash:

```
Workshop-2/
├──── src/
│     ├──── Homepage.java
│     ├──── News.java
│     ├──── Section.java
│     └──── InternationalNews.java
|──── Workshop-2.pdf
└──── README.md
```

Mockup:

-Mockup.

References:

-Viasure.

-Leanmind.