A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. Some nodes are highlighted with blue circles, and others with blue dots. The lines are thin and grey, creating a mesh-like structure.

# TDD + Code Coverage

Lorenzo Gabriel Pérez González alu0101233499  
Pablo Pérez González alu0101318318

A decorative network diagram in the bottom-right corner, similar to the one in the top-left, featuring a complex web of interconnected nodes and lines. Some nodes are highlighted with blue circles, and others with blue dots. The lines are thin and grey, creating a mesh-like structure.

# Team

**Pablo Pérez González**  
**alu0101318318**



**Lorenzo Gabriel Pérez González**  
**alu0101233499**

# Table of Contents

## 1. TDD

- a. Definition
- b. Pros
- c. Cons
- d. Conclusion

## 2. Code Coverage

- a. Definition
- b. Jest
- c. CodeCov
- d. Conclusion



Before we start...

This is really important.



A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by small circles, some of which are highlighted with a double-circle outline. The lines are thin and gray, creating a mesh-like structure.

# 1. TDD

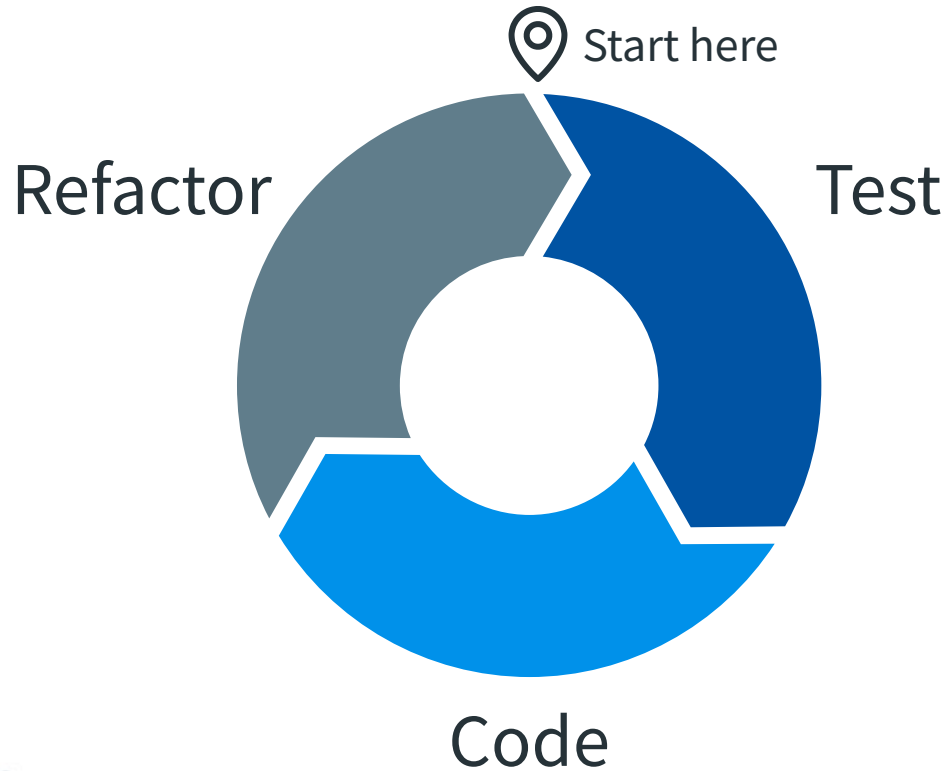
A decorative network diagram in the bottom-right corner, similar to the one in the top-left. It shows a cluster of nodes connected by lines, with some nodes having a double-circle outline. The overall style is minimalist and technical.

# Definition

**Test Driven Development** is a software development process that starts with the creation of a test and continues with the implementation of the code that makes it work.

```
1 describe('Sum', () => {  
2   test('Sum between two numbers', () => {  
3     expect(sumNumbers(2, 3)).toEqual(5);  
4   });  
5 });
```

# TDD's cycle







Why is TDD better than other  
software development processes?





“

***“Focus to a single feature at a time.”***

**Modularity**

A decorative network diagram at the top of the slide, featuring a series of interconnected nodes and lines. A central node is highlighted with a dashed circle and a blue double quote symbol.

“

***“Developers naturally produce a cleaner, more readable, and manageable code.”***

Maintenance

A decorative network diagram at the top of the slide, featuring a series of interconnected nodes and lines. The nodes are represented by small circles, some of which are highlighted with a blue outline. The lines are thin and gray, creating a web-like structure that spans the width of the slide. A central node is highlighted with a larger blue circle and a dashed border, containing a large blue quotation mark.

“

***“Modular improvement.”***

Refactoring

A decorative network diagram at the top of the slide, featuring a series of interconnected nodes and lines. A central node is highlighted with a dashed circle and a blue double quote icon.

“

***“TDD can reduce your  
time-to-market speed.”***

Decreasing costs



“

***“Tests act as documentation and  
illustrate how the code works.”***

Better documentation



“

***“TDD produces a higher overall  
test coverage”***

Less debugging



# What about TDD's cons?



A decorative network diagram at the top of the slide, featuring a series of interconnected nodes and lines. A central node is highlighted with a dashed circle and a blue double quote icon.

“

***“The team will be busy writing  
tests first.”***

Slow development

A decorative graphic at the top of the slide featuring a network of interconnected nodes and lines, resembling a molecular or digital structure. A central node is highlighted with a dashed circle and contains a large blue quotation mark.

“

***“Requires skills, persistence, and discipline.”***

Difficulty



“

***“Not every developer can make  
tests before having the code done”***

Strange approach

A decorative graphic at the top of the slide featuring a network of interconnected nodes and lines. A central node is highlighted with a dashed circle and a solid circle, containing a large blue quotation mark.

“

***“Tests could change to adapt.”***

Changing tests



TDD is easier and challenging to  
maintain?



# Tests $\neq$ Implementation

📌 Test maintenance is hard.

📌 Implementation maintenance is easy.





# Conclusions

# TDD: Conclusions

- **Code quality.**
- **Difficulty.**
- **Use in companies.**
- **Code coverage.**







## 2. Code Coverage

# Definition

Represents the percentage of code that has been tested.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	100	100	100	100	
nth-prime.js	100	100	100	100	

# Criteria



**Function Coverage:** Has each function been called?



**Statement Coverage:** Has each statement been executed?



**Edge Coverage:** Has the control flow been tested completely?



**Condition Coverage:** Has every condition been evaluated?



# Why is Code Coverage useful?





“

***“Higher code coverage finds  
more bugs”***

Quality



# How much Code Coverage is necessary?



# Nice Code Coverage



## ***Small projects***

- Aim to 100%



## ***Large projects***

- Aim to 70-80%



## ***High risk systems***

- Aim to the highest possible value.



# Code Coverage in Jest





# What is Jest?



## Jest



# Making a test

```
describe('Description of the tests', () => {  
  test('Description of the unit', () => {  
    expect(operation).toEqual(objective);  
    expect(operation).not.toEqual(objective);  
  });  
});
```

# Matchers



- 📌 **toBeGreaterThan**
- 📌 **toBeLessThan**
- 📌 **toBe**
- 📌 **toEqual**

- 📌 **toBeNull**
- 📌 **toBeDefined**
- 📌 **toBeUndefined**
- 📌 **toBeTruthy**
- 📌 **toBeFalsy**

- 📌 **toMatch**
- 📌 **toContain**
- 📌 **toThrow**



Now, how to do coverage with Jest?

# Noob mode

**Just add `--coverage`**

# Medium mode

Using a *package.json*

```
"scripts": {  
  "test": "jest"  
},  
"jest": {  
  "collectCoverage": true,  
  "collectCoverageFrom": ["./src/**"]  
},
```

# Pro mode

Using a good **package.json**.

```
"scripts": {  
  "test": "jest"  
},  
"jest": {  
  "collectCoverage": true,  
  "collectCoverageFrom": ["./src/**"],  
  "coverageThreshold": {  
    "global": {  
      "lines": 120  
    }  
  }  
},
```

# Pro mode

Using a good **package.json**.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	100	100	100	100	
division.js	100	100	100	100	
product.js	100	100	100	100	
subtract.js	100	100	100	100	
sum.js	100	100	100	100	
Jest: "global" coverage threshold for lines (120%) not met: 100%					





# Code Coverage in CodeCov



# What is CodeCov?



# Use CodeCov

1. **Give CodeCov permission to your Github.**
2. **Choose a repository (Token creation).**

# Use CodeCov

3. **Create the coverage directory with Jest.**
4. **Setup the CI with Github Actions.**

# Use CodeCov

5. **Download the CodeCov Uploader.**
6. **Upload coverage with `./codecov -t [Token]`**

A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by small circles, some of which are larger and have concentric circles, suggesting a hierarchical or central structure. The lines are thin and gray, connecting the nodes in a non-linear fashion.

# Conclusions

A decorative network diagram in the bottom-right corner, similar to the one in the top-left. It shows a cluster of nodes connected by lines, with some nodes being larger and more prominent than others, indicating a central or significant node in the network.

# Code Coverage: Conclusions

- **Code quality.**
- **Code safety.**



# Bibliography: TDD

1. **TDD Wikipedia:** [https://en.wikipedia.org/wiki/Test-driven\\_development](https://en.wikipedia.org/wiki/Test-driven_development)
2. **Learn TDD:** <https://github.com/dwyl/learn-tdd>
3. **Benefits of TDD:**  
<https://fortegrp.com/test-driven-development-benefits/#:~:text=Developers%20have%20less%20debugging%20to,quality%20of%20the%20final%20product>
4. **Software disasters:**  
<https://raygun.com/blog/costly-software-errors-history/>  
<https://www.rankred.com/biggest-software-failures/>



# Bibliography: Code Coverage

1. **Code Coverage Wikipedia:** [https://en.wikipedia.org/wiki/Code\\_coverage](https://en.wikipedia.org/wiki/Code_coverage)
2. **Why is Code Coverage important?:**  
<https://about.codecov.io/blog/who-cares-about-code-coverage-and-why/#:~:text=Code%20coverage%20is%20a%20simple,the%20quality%20of%20your%20code>
3. **Use Code Coverage with Jest:**  
<https://www.valentinog.com/blog/jest-coverage/>
4. **Jest's Code Coverage Documentation:**  
<https://jestjs.io/docs/configuration#coveragethreshold-object>
5. **CodeCov Quick Start:** <https://docs.codecov.com/docs>

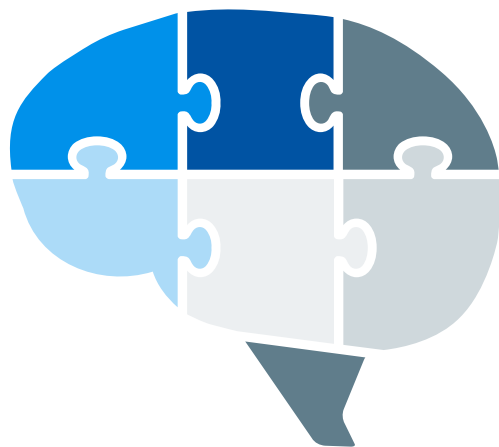
# Thanks!

## Any questions?

You can find us at:

[pablo.perez.gonzalez.23@ull.edu.es](mailto:pablo.perez.gonzalez.23@ull.edu.es)

[gabriel.perez.10@ull.edu.es](mailto:gabriel.perez.10@ull.edu.es)





## SlidesCarnival icons are editable shapes.

This means that you can:

- Resize them without losing quality.
- Change line color, width and style.

Isn't that nice? :)

Examples:



# Diagrams and infographics

