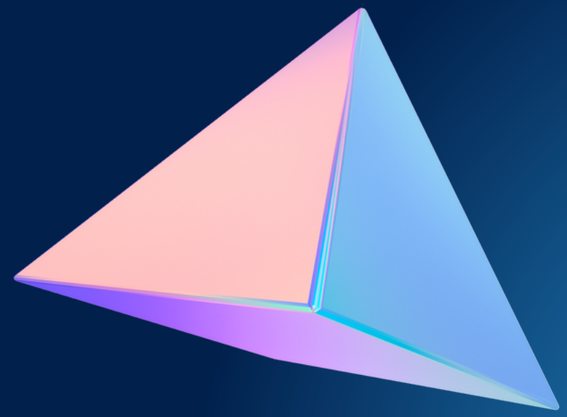
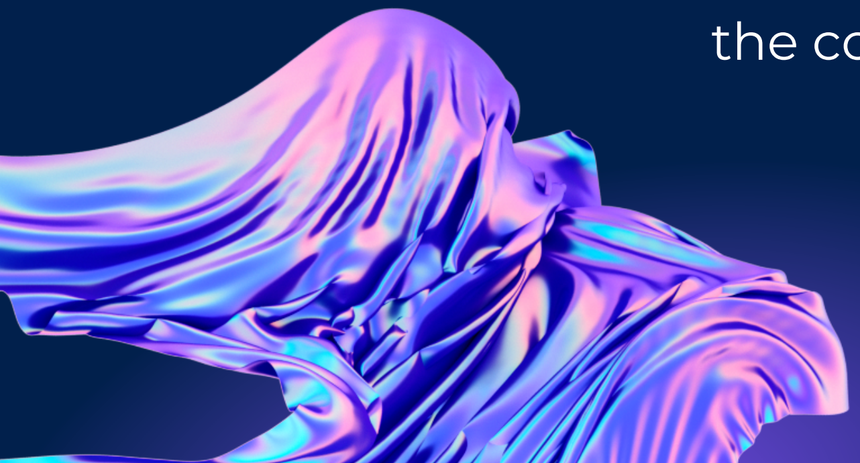
The Power of Code Reviews

DGL104 - Programming practice article



Introduction

Code reviews are like having a second set of eyes on the code before it goes live. They help catch mistakes, improve the overall quality of the code, and promote collaboration among team members



Benefits of Code Reviews

- Catch bugs early
- Improve code quality
- Promoting collaboration
- Maintaining coding standards

Catch bugs early

Code reviews are a great way to catch bugs early in the development process. By having peers review your code, potential issues can be spotted and fixed before they cause problems in the final product, saving time and effort in the long run

Improve code quality

Code reviews improve code quality by providing an opportunity for team members to share their knowledge and expertise. Through constructive feedback and discussions, code can be refined and optimized, leading to a higher-quality final product that meets both functional and non-functional requirements.

Promoting collaboration

Code reviews promote collaboration by encouraging team members to work together towards a common goal: creating high-quality software. Through code reviews, team members can share ideas, learn from each other, and build a sense of camaraderie that enhances the overall productivity and effectiveness of the team.

Maintaining coding standards

Code reviews help maintain coding standards by ensuring that all code written follows a set of guidelines or best practices. This consistency makes the code easier to read, understand, and maintain over time.

Best Practices for Code Reviews

- Define Goals
- Code Style and Standards
- Timing and Frequency
- Reviewer Selection
- Constructive Feedback
- Automation
- Follow-Up

Define Goals

Define goals in code reviews help ensure that the review process is focused and productive, guiding reviewers to look for specific aspects such as functionality, readability, and performance.

Code Style and Standards

Code style and standards refer to the guidelines and conventions that developers follow when writing code, ensuring consistency and readability across the codebase.

Timing and Frequency

Timing and frequency in code reviews refer to when and how often code reviews are conducted. This can vary depending on the team's workflow, but ideally, reviews should be done regularly and promptly after code changes are made to catch issues early.

Reviewer Selection

Reviewer selection is the process of choosing the right team members to review a piece of code based on their expertise and knowledge to provide valuable feedback.

Constructive Feedback

Constructive feedback in code reviews means providing comments and suggestions that help improve the code without being negative or discouraging. It focuses on pointing out areas for improvement and offering solutions or alternatives.

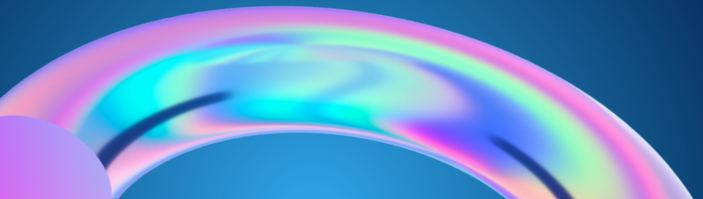
Automation

Automation in code reviews refers to using tools and scripts to automatically check for common issues, such as code formatting errors or potential bugs, saving time and improving code quality.

Follow-Up

Follow-up in code reviews involves addressing any feedback or issues identified during the review process. This may include making necessary changes to the code, discussing further with reviewers, or documenting decisions made for future reference.

Examples of Effective Code Reviews



Improving Readability

Before

```
for i in range(10):print(i)
```

After

```
for i in range(10):
    print(i)
```

Identifying Bugs

Before

```
int total = 0;
for (int i = 0; i < 5; i++) {
    total += i;
}
```

After

```
int total = 0;
for (int i = 1; i <= 5; i++) {
    total += i;
}
```

Reviewers noticed the off-by-one error in the loop condition, ensuring that the loop iterates five times as intended.

Ensuring Best Practices

Before

```
function calculateTotal(price, quantity) {
    return price * quantity;
}
```

After

```
function calculateTotal(price, quantity) {
    if (price < 0 || quantity < 0) {
        throw new Error('Price and quantity must be positive');
    }
    return price * quantity;
}
```

Reviewers suggested adding input validation to handle negative values, improving the robustness of the function.

Code Consistency

Before

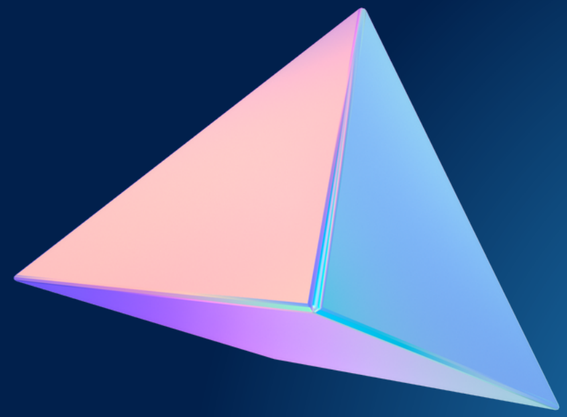
```
if condition:
    do_something()
```

After

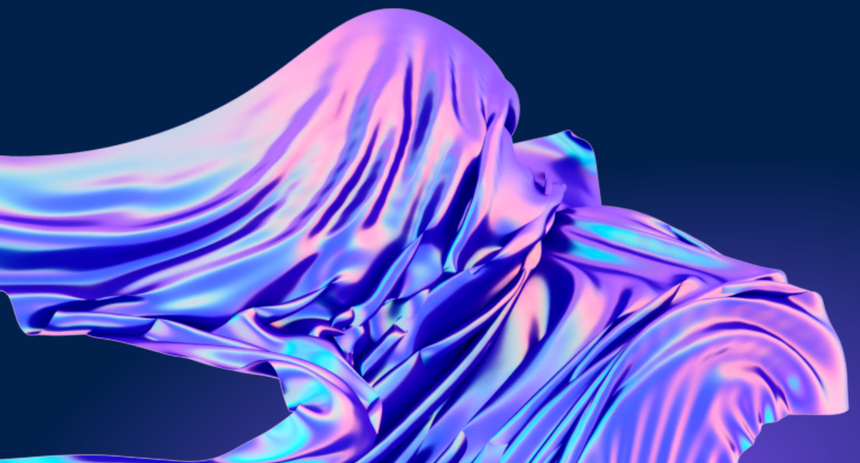
```
if condition:
    do_something()
else:
    do_something_else()
```

Reviewers pointed out the inconsistency in the code style, leading to a more uniform and easier-to-read structure.

Conclusion



Code reviews are essential for enhancing code quality and promoting teamwork. By providing valuable feedback, reviews improve readability, identify bugs, and ensure adherence to best practices. Implementing these practices in your workflow can lead to more reliable and maintainable code. Make code reviews a priority to create better software and foster a collaborative development environment.



Refrence

Refrence 1

Refrence 2

Refrence 3

Refrence 4

Thank You

Viren Bhadiyadra(N0200045)

vbhadiyadra@northislandcollege.ca