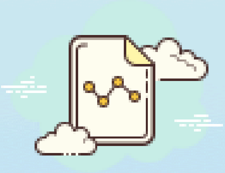
Post-Mortem Analysis and Continuous Improvement in DevOps

Group Presentation A2

Presented by: Group 2

Jan, 2025



Introduction

- **Objective**: Demonstrate how structured post-mortems lead to effective incident response and enhanced reliability
- **Agenda**:
  1. Company & Incident Overview
  2. Incident Response & Post-Mortem
  3. Root Causes & Lessons Learned
  4. Improvement Areas
  5. Recommended Corrective Actions
  6. Results & Impact

Incident Overview

- **XYZ Inc.**: Leading cloud-based platform for enterprise clients
- **Incident**: Routine system upgrade led to a misconfiguration error
- **Impact**:
  - Critical features became inaccessible
  - High customer dissatisfaction
  - Immediate rollback required to restore service

Incident Response & Post-Mortem

Immediate Response

- Rolled back to a stable version
- Engaged on-call DevOps team for swift resolution

Post-Mortem Steps

1. Cross-functional review (Dev, Ops, Support)
2. Root cause analysis (deployment script misconfiguration)
3. Blameless approach (learn more) to foster trust and transparency

Root Causes & Lessons Learned

Root Causes

1. Misconfiguration in deployment script
2. Inadequate change management
3. Communication gaps during the emergency

Lessons Learned

- Need comprehensive testing, including config checks
- Mandatory peer reviews and sign-offs
- Clear emergency communication protocols

⚠️

Improvement Areas

1. **Testing & Validation**

- Automated integration/end-to-end tests
- Regular “fire drills” to practice incident handling

2. **Change Management**

    - Peer reviews for production changes
    - Documented rollback procedures and approvals

3. **Communication & Collaboration**

    - Defined escalation paths (using a RACI model for clarity)
    - Standardized incident channels (Slack, Teams)

4. **Kanban Board as a Service**

    - Centralized backlog/task management for cross-functional teams
    - Real-time visibility of work in progress and bottlenecks
    - More ↓

Kanban: Making Work Flow Smoothly!

🎯 **How Kanban Helps Teams:**

- **See Everything at a Glance** 📋

  Tasks are clearly visualized in columns (To Do → In Progress → Done).

- **Stay Focused** 🎯

  Work-in-Progress (WIP) limits prevent overload and promote higher-quality results.

- **Find Bottlenecks Faster** 🚦

  Easily spot where work is stuck, so you can resolve issues quickly.

- **Work Together Better** 🤝

  Transparent workflows foster collaboration and ensure everyone's on the same page.

Recommended Corrective Actions

- **Infrastructure as Code** (IaC) & **Policy-as-Code [Terraform]** (PaC)
    - Automate checks for misconfigurations
- **Canary or Blue-Green Deployments** (Martin Fowler)
    - Test updates on a small subset to reduce risk
- **Enhanced Monitoring & Observability**
    - Proactive alerts, logs, and dashboards
- **Scheduled Retrospectives & Post-Mortems**
    - Continual improvement and knowledge sharing

Results & Impact

- **Reduced Downtime**: Faster rollback and detection
- **Improved Customer Satisfaction**: Fewer critical incidents and quicker resolutions
- **Stronger Team Dynamics**: Blameless culture & cross-functional collaboration
- **Key Metrics** [DevOps Research and Assessment (DORA)]:
  - **Deployment Frequency**
  - **Lead Time for Changes**
  - **Change Failure Rate**
  - **Time to Restore Service**

Thank You

Thank you for your time and attention!

🤝