**DevOps Project Presentation:**

**Presentation Title**

* **DevOps Project Name : Make Your Wishes Come True**
* **Participants Name :**
  + Aymane EL HAMRAOUI
  + Pape Mamadou DIAGNE
  + Chair mehdi
* **Date : January 26, 2025**

**1. Introduction**

**Context:**

The project addresses a common challenge: people's difficulty in maintaining New Year's resolutions. Traditional resolution tracking suffers from:

* Manual tracking leading to inconsistent monitoring
* Lack of integrated reminders and progress updates
* Poor user engagement and motivation retention
* No automated deployment process for updates and features
* Limited collaboration between UI/UX teams and backend operations

DevOps was chosen because it enables:

* Continuous integration of new features (like monthly challenges)
* Rapid deployment of motivational content updates
* Automated testing of user engagement features
* Seamless integration between mobile app and smartwatch components

**Objectives :**

1. **Technical Objectives :**

* Implement automated CI/CD pipeline for mobile and smartwatch apps
* Deploy updates without service interruption
* Maintain 99.9% uptime for notification services
* Reduce deployment time from 2 days to 2 hours

1. **Business Objectives :**

* Deliver monthly resolution themes consistently
* Enable rapid user feedback integration
* Support scaling from 100 to 10,000 users
* Maintain user engagement through automated features

**2. Architecture and Workflow**

* **Architecture Diagram:**

A screenshot of a computer

Description automatically generated

**Tools Used at Each Stage**

1. **Source Control:**

* **GitHub for version control**
* **Branch protection rules**
* **Pull request reviews**

1. **CI/CD :**

* **Jenkins for pipeline automation**

**General Workflow**

1. **Code Commit :** 
   * **Feature branches**
   * **Pull request creation**
   * **Code review process**
2. **Automated Build/Testing:** 
   * **Unit tests**
   * **Integration tests**
   * **Security scans**
3. **Deployment:** 
   * **Automated staging deployment**
   * **Manual production approval**
   * **Rolling updates**
4. **Monitoring :** 
   * **Performance metrics**
   * **User engagement tracking**
   * **Error logging**

**3. Tools and Technologies**

**Core Tools Selected**

1. **Source Control Management :**
   * GitHub : Primary code repository
   * Git : Version control system
2. **CI/CD:** 
   * Jenkins : Pipeline automation & orchestration
   * GitHub Actions: Secondary automation for specific tasks
3. **Testing:** 
   * Jest: Frontend unit testing
   * PyTest: Backend testing
   * Cypress: E2E testing
4. **Deployment & Infrastructure :** 
   * Docker : Containerization
   * AWS : Cloud infrastructure
   * Kubernetes: Container orchestration

**Selection Justification**

1. **GitHub + Jenkins Integration:**
   * Strong webhook integration
   * Extensive plugin ecosystem
   * Familiar to team members
   * Free for open source projects
2. **Testing Stack :** 
   * Jest: React Native compatibility
   * PyTest: Python backend support
   * Cypress: Mobile app testing support
3. **Infrastructure :**
   * Docker : Consistent environments
   * AWS : Scalability needs
   * Kubernetes: Microservices architecture

**4. CI/CD Pipeline**

* **Main Stages Description**
  + 1. **Build Stage**
  + Fetches code from GitHub
  + Installs dependencies using npm
  + Compiles React application
  + Creates Docker container
  + Stores artifacts
    1. **Testing Stage**
  + Unit tests with Jest
  + Integration tests
  + End-to-end tests with Cypress
  + Code coverage reports
    1. **Deployment Stage**
  + Deploying to dev/staging/prod environments
  + Rolling updates
  + Environment-specific configurations
    1. **Validation Stage**
  + Health checks
  + Performance monitoring
  + User journey tests
  + Security scans

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

1. **Sample YAML or Jenkinsfile pipeline configuration :**

pipeline {

agent any

environment {

DOCKER\_IMAGE = 'resolution-app'

DOCKER\_TAG = "${BUILD\_NUMBER}"

}

parameters {

choice(name: 'ENV', choices: ['dev', 'staging', 'prod'])

}

stages {

stage('Build') {

steps {

sh 'npm install'

sh 'npm run build'

sh "docker build -t ${DOCKER\_IMAGE}:${DOCKER\_TAG} ."

}

}

stage('Test') {

parallel {

stage('Unit') {

steps {

sh 'npm run test:unit'

}

}

stage('Integration') {

steps {

sh 'npm run test:integration'

}

}

}

}

stage('Deploy') {

steps {

script {

if (params.ENV == 'prod') {

input 'Deploy to production?'

}

sh "./deploy.sh ${params.ENV}"

}

}

}

stage('Validate') {

steps {

sh './health-check.sh'

sh './smoke-tests.sh'

}

}

}

post {

always {

junit '\*\*/test-results/\*.xml'

}

}

}

**5. Demonstration**

**Pipeline Execution Demo**

1. **Code Commit Trigger**

* Push code to feature branch
* Create pull request
* Show automated PR checks

1. **Jenkins Pipeline Visualization**

* Display pipeline stages execution
* Show parallel test runs
* Highlight build artifacts

1. **Test Results Dashboard**

* Unit test coverage: 85%
* Integration tests passed: 45/45
* End-to-end scenarios: 12/12 passing

1. **Deployment Process**

* Environment selection prompt
* Configuration validation
* Deployment logs
* Health check results

1. **Monitoring Dashboard**

* Application performance metrics
* Error rates
* User engagement statistics
* Resource utilization

**Final Application Demo**

* Mobile app interface
* Smartwatch integration
* Monthly goals tracking
* User notification system
* Progress analytics

**6. Conclusion**

**Benefits Achieved Through DevOps**

1. **Process Improvements**

* Deployment time reduced from 2 days to 2 hours
* Test coverage increased to 85%
* Zero downtime deployments achieved
* Automated security checks implemented

1. **Team Impact**

* Cross-functional collaboration improved
* Release frequency increased
* Bug detection earlier in development cycle
* Knowledge sharing enhanced

**Future Outlook**

1. **Immediate Plans**

* Implement auto-scaling
* Enhance monitoring metrics
* Add performance testing
* Expand test automation

1. **Long-term Vision**

* Adopt Infrastructure as Code
* Implement chaos engineering
* Scale to multiple regions
* Share DevOps practices across teams

.

**7. Appendices**

**Git link :** [**https://github.com/Pape45/wishes-app**](https://github.com/Pape45/wishes-app)

**User Documentation**

**How to Use the Resolution Tracking Pipeline**

1. **Basic Usage**

* **Select month from dropdown**
* **Enter current day (1-31)**
* **Check/uncheck tasks as completed**
* **Click "Build" to generate progress report**

1. **Task Descriptions by Month**

* **Each month has 3 specific tasks**
* **Tasks automatically update when month changes**
* **Progress tracked via checkboxes**

1. **Reports**

* **HTML reports generated automatically**
* **Shows progress bar and task status**
* **Archived for future reference**