# Feature Justification & Jira Integration

## 1. Why These Features Were Chosen?

Our feature selection for project timeline and defect prediction was based on logical relevance, data-driven analysis, and industry best practices. We used statistical methods like correlation analysis and feature importance ranking from machine learning models to confirm these selections.

### 1.1 Project Timeline Prediction Features

• team\_size: More members can speed up a project but also increase coordination challenges.

• task\_count: More tasks increase workload, extending project duration.

• developer\_experience: Experienced developers work more efficiently, reducing delays.

• priority\_level: High-priority projects receive more resources, affecting completion time.

• task\_complexity: More complex tasks require more effort and take longer.

• project\_size: Larger projects require more development time.

• testing\_coverage: More testing ensures quality but increases effort and duration.

• Effort\_Density: Indicates how much effort is required relative to the project size.

• Team\_Productivity: More productive teams can complete tasks faster.

• LoC\_per\_Team\_Member: More lines of code per developer can indicate efficiency but also riskier changes.

### 1.2 Defect Prediction Features

• defect\_fix\_time\_minutes: Longer fix times indicate complex issues and potentially more defects.

• size\_added: More added code increases defect probability.

• size\_deleted: Removing large sections of code might introduce instability.

• size\_modified: Frequent modifications increase the chance of defects.

• effort\_hours: More effort may indicate complexity, leading to higher defect rates.

• complexity\_score: More complex components tend to have more defects.

• testing\_coverage: Better test coverage reduces defect count but increases effort.

• team\_key (encoded): Some teams may have higher defect rates due to skill gaps.

By selecting these features, we ensured that our models are grounded in software engineering principles, statistical validation, and machine learning feature importance analysis.

## 2. How These Features Relate to Jira & Other Project Management Tools

Many of our selected features align well with project management tools like Jira, Trello, and Azure DevOps. These tools track project progress, productivity, issue tracking, and defect resolution, which directly influence software project timelines and defect counts.

### 2.1 Matching Features in Jira

Here’s how our features match with Jira and other tools:

• team\_size: Equivalent to Assignee Count in Project/Task in Jira.

• task\_count: Equivalent to Number of Jira Issues (Tickets) in Jira.

• developer\_experience: Equivalent to Developer Skill Level (Not explicitly tracked in Jira) in Jira.

• priority\_level: Equivalent to Task Priority (High, Medium, Low) in Jira.

• task\_complexity: Equivalent to Story Points / Task Complexity in Jira.

• project\_size: Equivalent to Epic Size / Total Story Points in Jira.

• testing\_coverage: Equivalent to Test Execution Status (via plugins like Zephyr, Xray) in Jira.

• Effort\_Density: Equivalent to Work Log / Effort Logged Per Task in Jira.

• Team\_Productivity: Equivalent to Velocity / Sprint Burndown Reports in Jira.

• LoC\_per\_Team\_Member: Equivalent to Code Changes Per Developer (via Git Integrations) in Jira.

• defect\_fix\_time\_minutes: Equivalent to Average Time to Resolve Issues (MTTR) in Jira.

• size\_added, size\_deleted, size\_modified: Equivalent to Code Changes (via Git Integrations) in Jira.

• effort\_hours: Equivalent to Work Log Hours / Time Tracking in Jira.

• complexity\_score: Equivalent to Issue Complexity (via Story Points) in Jira.

• team\_key (encoded): Equivalent to Team Assignment / Workload Per Team in Jira.

### 2.2 Integrating with Jira

Some features can be directly obtained from Jira, while others require integrations with Git (Bitbucket, GitHub, GitLab) and test management tools (Zephyr, Xray). Our approach ensures that predictions are based on real-world project tracking data, making them practical and applicable.