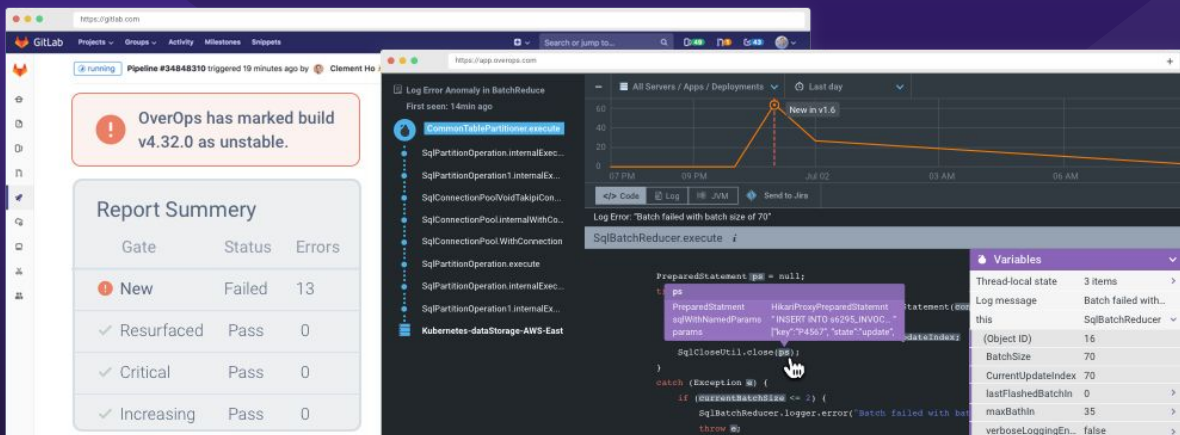


# Enhance Your CI/CD Pipeline with Runtime Code Analysis from OverOps

Block a release when critical errors are detected, and get the complete context required to resolve



## Surface Runtime Errors Directly in GitLab

Mission-critical applications are expected to deliver a reliable customer experience. But today, when speed is essential in order to remain competitive – companies are introducing critical errors in up to 60% of their new code releases, exposing the limitations of testing and static analysis, driving the need for a new approach.

OverOps is a continuous reliability solution that analyzes code as it executes to identify critical errors and resolve them in minutes - even if they weren't tested for or logged.

**Identify** new, critical, resurfaced and unique runtime errors in each release – without relying on foresight.

**Prevent** unreliable releases from being deployed into UAT and production with automated quality gates.

**Resolve** issues with complete code variables, DEBUG logs and host / container state at the moment of event.

# Ensure Your Code is Production-Ready

Introducing Runtime Quality Gates:



New Errors



Resurfaced Issues



Critical Exception Types



Unique Errors

## Block a Release When New Errors Are Detected

With OverOps runtime quality gates

- Use out-of-the-box quality criteria or customize your own
- Prioritize reliability issues by their severity
- Drill down into critical errors to quickly resolves

OverOps has marked build v4.32.0 as unstable.

Report Summary		
Gate	Status	Errors
New	Failed	13
Resurfaced	Pass	0
Critical	Pass	0
Increasing	Pass	0

```
ps
PreparedStatement HikariProxyPreparedStatement
sqlWithNamedParams "INSERT INTO s6295_INVO...
params [{"key":"P4567","state":"update"}]

sqlCloseUtil.close(B@);
}
catch (Exception e) {
  if (currentBatchSize <= 2) {
    SqlBatchReducer.logger.error("Batch failed
    throw e;
  }
  this.maxBatchSize = currentBatchSize / 2;
  logger.ERROR("Batch failed with batch size of
  this.maxBatchSize, e.getMessage());
  SqlBatchReducer.execute i
}

public static boolean insertInvCounts(final Stri
TakipiConnection conn) {
  if (shouldInsertInvCount(serviceId)) {
```

Variables

- Thread-local state 3 Items
- Log message Batch failed with...
- this SqlBatchReducer
- conn TakipiConnection
- currentBatchSize 70
- e BatchUpdateExc...
- namedParamsBatch Object [70]
- ps TakipiPreparedSt...
- serviceId "S6295"
- sqlTemplateParams "/sql/invocacion...
- templateParams HashMap[7]

Variables

- conn TakipiConnection
- (Object ID) 17

## Get the Complete Context Required to Resolve

With error snapshots from OverOps

- See the stack trace, source code and variable state for every error
- Access relevant TRACE and DEBUG level logs, even if they were turned off
- Capture detailed host/container state from the moment of error