

1 Start Workers

```
queuctl worker start --count 2
```

```
[worker-1]
```

```
[worker-2]
```

```
[worker-1] waiting for jobs...
```

```
[worker-2] waiting for jobs...
```

2 Basic Successful Job

```
queuctl enqueue -c "echo hello" --id ok1
```

```
id: ok1
```

```
command: echo hello
```

```
state: completed
```

```
attempts: 0
```

3 Failing Job → Retries → DLQ

```
queuctl enqueue -c 'node -e "process.exit(1)"' --id fail1
```

```
id: fail1
```

```
command: node -e "process.exit(1)"
```

```
state: dead
```

```
attempts: <max_retries>
```

4 Retry a DLQ Job

```
queuctl dlq retry fail1
```

```
id: badcmd1
```

```
state: dead
```

```
reason: command not found
```

```
attempts: <max_retries>
```

6 Multiple Concurrent Jobs

```
for i in 1 2 3 4 5
do
  queuectl enqueue -c "echo $i" --id job$i
done
```

```
id: j1, state: completed
id: j2, state: completed
id: j3, state: completed
id: j4, state: completed
id: j5, state: completed
```

Edges cases

1 Enqueueing a job with a duplicate ID

```
queuectl enqueue -c "echo hi" --id ok1
```

Error: Job with id ok1 already exists

2 Invalid JSON Payload

```
queuectl enqueue '{"id": "x", "command":}'
```

Error: Invalid job payload (JSON parse failed)

3 Stopping workers while jobs are pending

1. Enqueue 10 slow jobs:

```
for i in {1..10}; do queuectl enqueue -c "sleep 3" --id slow$i;
done
```
2. Start workers:

```
queuectl worker start --count 2
```

3. Immediately stop:
`queuctl worker stop`

Final Full Test in Order

You can paste this as a single section:

```
queuctl worker start --count 2
queuctl enqueue -c "echo ok" --id ok1
queuctl enqueue -c "node -e 'process.exit(1)'" --id fail1
queuctl enqueue -c "unknowncmd" --id badcmd1
```

```
queuctl list --state pending
queuctl status
queuctl dlq list
queuctl dlq retry fail1
queuctl status
queuctl worker stop
```