

# LittleLives Sr. Fullstack Engineer - Interview Answer Document

Nguyễn Nam Long - 1997 - [j2teamnnl@gmail.com](mailto:j2teamnnl@gmail.com)

## Phần 1: Xử Lý Vấn Đề Kỹ Thuật (Technical Problem Solving)

### 1.1 Chẩn Đoán Ngay Lập Tức (Immediate Diagnosis)

#### Hypothesis về Root Cause

Dựa trên timeline và các thay đổi gần đây, tôi nghi ngờ **tính năng attendance tracking mới deploy hôm qua** là nguyên nhân chính:

- Monitor chỉ báo lỗi sáng nay, không phải cả tuần → timing trùng với deploy hôm qua
- Database connections gần cạn (95/100) → có thể N+1 query problem
- Response time 15s → query không efficient hoặc blocking operations

Thêm vào đó, real-time notifications (tuần trước) kết hợp với database tăng 40% (pilot Indonesia) cũng góp phần làm trầm trọng thêm vấn đề.

#### 5 Logs/Metrics Kiểm Tra Đầu Tiên

#	Log/Metric	Lý do
1	Slow query logs	Xác định query nào đang chậm
2	APM traces (Laravel Telescope/Clockwork)	Xem breakdown time và số query trong request
3	Git history của attendance feature	Xác định thay đổi cụ thể và người phụ trách
4	Database connection pool metrics	Confirm connection exhaustion
5	Error logs của attendance service	Xem lỗi cụ thể đang xảy ra

#### Team Member Gọi Đầu Tiên

Tôi sẽ gọi **developer đã deploy attendance feature hôm qua** (dựa theo git history):

- Họ hiểu rõ nhất về thay đổi gần đây
- Có thể pair program để fix nhanh

- Có context để rollback nếu cần

## 1.2 Phân Tích Code (Code Analysis)

### 3 Performance Problems Đã Xác Định

#### Problem 1: N+1 Queries

- Code đang query lần lượt: Child → School → Attendance → Parents → COUNT → Update
- Mỗi request có 6+ queries, với nhiều học sinh sẽ tăng exponentially

#### Problem 2: Synchronous Notifications

- Notifications đang chạy sync trong request cycle
- User phải đợi tất cả notifications gửi xong mới nhận response
- Đây không phải core feature (điểm danh), không cần blocking

#### Problem 3: Expensive COUNT Query

- Mỗi lần điểm danh đều chạy COUNT cho cả trường
- Query này full table scan, rất expensive khi data lớn
- Thực tế chỉ cần increment counter

### Code Cải Tiến

#### Version 2 - Hotfix (Deploy ngay):

```
public function markAttendance($childId, $status)
{
    $child = $this->Child->findById($childId);
    $school = $this->School->findById($child['Child']['school_id']);
    $parents = $this->Parent->findByChildId($childId);

    $this->Attendance->save([
        'child_id' => $childId,
        'school_id' => $school['School']['id'],
        'status' => $status,
        'timestamp' => date('Y-m-d H:i:s'),
        'marked_by' => $this->Auth->user('id')
    ]);

    // Đẩy notifications vào job (async)
    $this->dispatch(new SendAttendanceNotificationsJob([
        'child' => $child,
        'status' => $status,
        'parents' => $parents
    ]));
}
```

```

// Dùng increment thay vì SELECT + UPDATE
$this->School->increment($school['School']['id'], 'today_attendance_count');

return $this->redirect('/attendance/success');
}

```

### Version 3 - Eager Loading:

```

public function markAttendance($childId, $status)
{
    // 1 query với eager loading
    $child = $this->Child->find('first', [
        'conditions' => ['Child.id' => $childId],
        'contain' => ['School', 'Parent']
    ]);

    if (!$child) {
        throw new NotFoundException('Child not found');
    }

    $this->Attendance->save([
        'child_id' => $childId,
        'school_id' => $child['School']['id'],
        'status' => $status,
        'timestamp' => date('Y-m-d H:i:s'),
        'marked_by' => $this->Auth->user('id')
    ]);

    $this->dispatch(new SendAttendanceNotificationsJob([
        'child' => $child,
        'status' => $status,
        'parents' => $child['Parent']
    ]));

    $this->School->increment($child['School']['id'], 'today_attendance_count');

    return $this->redirect('/attendance/success');
}

```

## Version 4 - Bulk Attendance (Thay đổi cơ chế):

```
public function markBulkAttendance($schoolId, array $childIds, $status)
{
    // Query school + tất cả children + parents 1 lần
    $school = $this->School->find('first', [
        'conditions' => ['School.id' => $schoolId],
        'contain' => [
            'Child' => [
                'conditions' => ['Child.id' => $childIds],
                'Parent'
            ]
        ]
    ]);

    if (!$school || empty($school['Child'])) {
        throw new NotFoundException('School or children not found');
    }

    $timestamp = date('Y-m-d H:i:s');
    $markedBy = $this->Auth->user('id');

    // Bulk insert attendance records
    $attendanceRecords = array_map(fn($child) => [
        'child_id' => $child['id'],
        'school_id' => $schoolId,
        'status' => $status,
        'timestamp' => $timestamp,
        'marked_by' => $markedBy
    ], $school['Child']);

    $this->Attendance->saveMany($attendanceRecords);

    // 1 job cho tất cả notifications
    $this->dispatch(new SendBulkAttendanceNotificationsJob([
        'school' => $school,
        'children' => $school['Child'],
        'status' => $status
    ]));

    // Increment 1 lần theo số lượng
    $this->School->incrementBy($schoolId, 'today_attendance_count', count($childIds));

    return $this->redirect('/attendance/success');
}
```

### 1.3 Chiến Lược Fix Nhanh (Quick Fix Strategy)

1. Deploy Version 2 ngay:

- Đẩy notifications vào job (không block request)
- Thay COUNT + UPDATE bằng increment
- Không sửa logic core, chỉ move code → an toàn

2. Nếu vẫn chưa ổn:

- Stats update → chuyển sang cron job cuối ngày

3. Traffic Routing Strategy:

- Route Indonesian traffic sang server riêng (nếu có)
- Rate limiting cho attendance endpoint để tránh spam
- Scale up database connections nếu cần

4. Rollback Plan:

- T+0: Deploy hotfix
- T+5: Check metrics
- T+10: Nếu không cải thiện → git revert về version trước
- T+15: Verify rollback thành công
- T+20: Stable cho demo và testing

## Phần 2: Quyết Định Kiến Trúc & Leadership

### 2.1 Đánh Giá Rủi Ro Kỹ Thuật (Technical Risk Assessment)

Xếp Hạng Rủi Ro

Rank	Rủi Ro	Reasoning
1	Cross-region latency (Indonesia)	Ảnh hưởng trực tiếp trải nghiệm khách hàng. Công ty product → khách hàng là ưu tiên số 1. Mất khách = mất doanh thu.
2	Database bottleneck	Database treo → mọi thứ đều chết. Ảnh hưởng toàn bộ hệ thống.
3	Mobile app crash	Ảnh hưởng UX nhưng có thể fix bằng rate limiting. Khách chờ 1-2 giây được, nhưng crash thì không.

Rank	Rủi Ro	Reasoning
4	PHP memory limits	Có thể tăng memory (tốn tiền nhưng nhanh). Khách tăng → doanh thu tăng → cover được chi phí.
5	AWS costs	Quan trọng về business nhưng không phải technical blocker. Đang có khách = đang có doanh thu.

Mitigation Strategy cho #1 (Cross-region Latency)

Short-term (ngay):

- CDN cho static assets (CloudFront endpoint Indonesia)
- Giảm payload size trong API responses
- Cache frequently accessed data (Redis)

Medium-term (1-2 tháng):

- Deploy read replica tại Indonesia region

Long-term:

- Multi-region deployment
- Data locality compliance

2.2 Tình Huống Phân Bố Nguồn Lực (Team Resource Dilemma)

Recommendation

Giả sử team có 6 người:

Tuần	Option C (Monitoring)	Option B (Localization)	Option A (Performance)
1	2 người ✓	4 người (in progress)	-
2	Done → chuyển A	4 người (demo ready)	2 người
3+	-	Tiếp tục	2 người ongoing

Lý do:

1. **Option C trước** (2 người, 1 tuần): Monitoring/alerting là foundation. Không có thì khách gặp vấn đề → có thể đi luôn mà mình không biết.
2. **Option B song song** (4 người): Tính năng cho Indo là commitment với khách. 2 tuần phải có bản demo để show progress.

3. **Option A sau** (2 người từ C): Performance issues luôn tồn tại trong product company. 2 người có thể fix những vấn đề critical trước.

## Communicate Với CEO




"Tôi hiểu urgency của việc onboard 500 trường Indonesia.

**Thực tế:** Không thể ship cả 3 với full quality trong 2 tuần với team hiện tại.

**Đề xuất của tôi:**

- Tuần 1: Monitoring (2 người) + Localization bắt đầu (4 người)
- Tuần 2: Localization có bản demo + Performance fix (2 người)

**Kết quả sau 2 tuần:**

-  Monitoring/alerting system hoàn thành
-  Bản demo Indonesian features để show khách
-  Bắt đầu fix performance issues

## Nói Với Team Members

"Tôi biết option A (performance) cũng rất quan trọng.

**Lý do defer:**

- Monitoring cần làm trước để phát hiện vấn đề sớm
- Indonesian features là commitment với khách đang chờ

**Commitment của tôi:**

- Sau tuần 1, 2 người sẽ chuyển sang performance
- Performance issues sẽ được track và prioritize
- Đây là ongoing effort, không phải bỏ qua

**Note:** Đông người ≠ xong nhanh. Dồn cả team vào 1 task có thể gây conflict và communication overhead."

## Sử Dụng AI (AI Usage)

### Đã Dùng AI Cho:

- Tóm tắt và dịch đề bài sang tiếng Việt để hiểu rõ hơn
- Tạo flow hướng dẫn trả lời
- Syntax checking và code formatting

- Tạo document này

## ✖ Tự Phân Tích:

- Xác định root cause từ timeline và experience
- Thiết kế các version fix (V2 → V3 → V4)
- Risk assessment dựa trên kinh nghiệm làm product company
- Priority decisions và communication strategy

## Validation:

- Review code AI generate để đảm bảo logic đúng
- Cross-check với experience làm hệ thống điểm danh thực tế
- Đảm bảo giải pháp practical, không over-engineering
- Kiểm soát và hiểu được AI đã làm gì

## Kết Luận

---

Đây là lần đầu tiên tôi làm dạng challenge interview như thế này và thấy rất ấn tượng. Bài test reflect rất tốt các scenario thực tế mà Sr. Engineer sẽ gặp.

Điều tôi thích về LittleLives:

- Bài toán technical thực tế, không phải coding puzzle
- Focus cả leadership và communication, không chỉ code
- Cách tiếp cận cho phép sử dụng AI một cách honest

Tôi tin rằng với kinh nghiệm làm product company và từng build hệ thống quản lý trường học, tôi có thể contribute effectively vào team LittleLives.