HƯỚNG DẪN CHUẨN BỊ BẢO VỆ ĐỒ ÁN

"Từ A đến Z để tự tin trước hội đồng"

PHẦN 1: TÂM LÝ VÀ TƯ DUY

1.1. Hiểu về hội đồng

Họ KHÔNG muốn:

- Đánh trượt bạn
- Hỏi xoáy để bạn không trả lời được
- Tìm lỗi sai trong code

Họ MUỐN:

- Biết bạn có HIỂU những gì mình làm không
- Biết bạn có khả năng TƯ DUY và GIẢI QUYẾT VẤN ĐỀ không
- Biết bạn có thể GIẢI THÍCH kỹ thuật cho người khác không

> **Mindset đúng:** Hội đồng là người muốn nghe bạn chia sẻ về "đứa con tinh thần" của mình.

1.2. Bạn là CHUYÊN GIA

Về project CineBook này:

- Bạn đã code từng dòng
- Bạn hiểu từng quyết định
- Bạn biết từng bug đã fix

Không ai hiểu project này hơn bạn.

Hội đồng chỉ nhìn vào trong 15-20 phút. Bạn đã sống với nó hàng tháng.

PHẦN 2: CÂU HỎI "BẪY" VÀ CÁCH XỬ LÝ

2.1. "Tại sao không dùng X thay vì Y?"

Ví dụ: "Tại sao dùng Laravel mà không dùng Node.js?"

ĐỪNG nói:

- "Em không biết Node.js"
- "Laravel dễ hơn"

HÃY nói:

```
"Em đã cân nhắc giữa Laravel và Node.js. Em chọn Laravel vì:
1. Ecosystem hoàn chỉnh cho web app (Eloquent ORM, Blade templating)
2. Convention over configuration - phù hợp với timeline dự án
3. Cộng đồng Việt Nam lớn, tài liệu tiếng Việt nhiều

Nếu dự án cần real-time nhiều hơn (như chat), Node.js có thể phù hợp hơn.
Nhưng với Cinema Booking, Laravel đáp ứng tốt các yêu cầu."
```

Pattern trả lời:

1. Thừa nhận đã cân nhắc các option
2. Nêu lý do cụ thể cho lựa chọn
3. Acknowledge trường hợp option kia tốt hơn

2.2. "Nếu có 1 triệu user thì sao?"

ĐỪNG nói:

- "Dạ em chưa nghĩ đến"
- "Chắc sẽ chậm ạ"

HÃY nói:

```
"Với kiến trúc hiện tại, em đã chuẩn bị cho scalability:

1. Database level:
   - Đã đánh index cho các trường query thường xuyên
   - Sử dụng Eager Loading tránh N+1
   - Có thể thêm Read Replica khi cần

2. Application level:
   - Stateless design - có thể chạy nhiều instance
   - Session lưu database, dễ chuyển sang Redis
   - Cache layer có thể thêm vào

3. Nếu cần scale thực sự:
   - Tách microservices (Booking service, User service)
   - Message queue cho async tasks
   - CDN cho static assets

Em thiết kế với mindset 'scale khi cần', không over-engineer từ đầu."
```

2.3. "Code này có vấn đề bảo mật gì không?"

ĐỪNG nói:

- "Em nghĩ là an toàn rồi ạ"
- "Em không thấy vấn đề gì"

HÃY nói:

```
"Em đã implement các lớp bảo mật:

1. Authentication: Laravel's built-in auth với bcrypt
2. Authorization: Middleware role:admin cho admin routes
3. CSRF: Token tự động trên mọi form
4. SQL Injection: Eloquent ORM với prepared statements
5. XSS: Blade auto-escaping với {{ }}

Những điểm em biết CẦN cải thiện:
- Chưa có rate limiting cho API
- Chưa implement 2FA cho admin
- Nên thêm audit log cho sensitive actions

Em đã document những điểm này trong roadmap phát triển."
```

Key: Thể hiện bạn BIẾT giới hạn của mình.

2.4. "Em tự code hay copy từ đâu?"

HÃY nói:

```
"Em tự viết logic nghiệp vụ. Cụ thể:

Tự code 100%:
- Thuật toán tạo sơ đồ ghế
- Logic tính giá vé động
- QR check-in flow
- Dashboard analytics

Sử dụng và customize:
- Laravel framework (không ai viết lại framework)
- Bootstrap cho UI (có customize màu sắc, layout)
- Chart.js cho biểu đồ

Học và áp dụng:
- MVC pattern từ documentation
- Best practices từ Laravel community

Em tin rằng biết khi nào dùng tool có sẵn và khi nào tự code
là kỹ năng quan trọng của developer."
```

2.5. "Phần nào em thấy khó nhất?"

Đây là cơ hội VÀNG để thể hiện!

```
"Phần em thấy thử thách nhất là xử lý RACE CONDITION khi đặt ghế.

Vấn đề: 2 người cùng đặt 1 ghế cùng lúc.

Lần đầu em code:
- Chỉ check ghế trống rồi insert
- Bug: Cả 2 đều đặt được cùng 1 ghế!

Giải pháp em tìm ra:
- Database Transaction với row locking
- Check và insert trong cùng transaction
- Unique constraint ở database level

Bài học: Không chỉ code chạy đúng trên máy mình,
mà phải nghĩ đến concurrent users."
```

Pattern: Kể như một câu chuyện - Vấn đề → Thất bại → Học hỏi → Giải quyết

---

PHẦN 3: DEMO SCRIPT CHI TIẾT

3.1. Timeline 7 phút demo

```
[0:00-0:30] MỞ ĐẦU
"Đây là CineBook - hệ thống quản lý rạp chiếu phim.
Em sẽ demo 2 phần: Admin Panel và User Booking Flow."

[0:30-3:30] ADMIN PANEL (3 phút)
- Dashboard (30s): Chỉ các KPI, chart
- Quản lý phim (45s): Thêm 1 phim mới, gắn thể loại
- Quản lý phòng (45s): Xem sơ đồ ghế, giải thích 3 loại ghế
- Quản lý suất chiếu (45s): Tạo suất chiếu, chỉ công thức giá
- Quản lý booking (15s): Filter, xem chi tiết

[3:30-6:00] USER FLOW (2.5 phút)
- Chọn phim (30s)
- Chọn suất chiếu (30s)
- Chọn ghế - ĐIỂM NHẤN (60s): Real-time, màu sắc, giá
- Thanh toán (30s)

[6:00-6:30] QR CHECK-IN (30s)
- Scan QR
- Hiện thông tin vé
- Check-in thành công

[6:30-7:00] KẾT THÚC
"Đó là những tính năng chính của CineBook.
Em sẵn sàng trả lời câu hỏi của thầy cô."
```

3.2. Chuẩn bị data demo

```
PHẢI CÓ SẴN:
☑ 5-10 phim với poster đẹp
☑ 3-4 phòng chiếu với sơ đồ ghế
☑ 10-20 suất chiếu (quá khứ và tương lai)
☑ 20-30 booking với các trạng thái khác nhau
☑ 5-10 user (có 1 admin, còn lại user thường)
☑ Dashboard có số liệu đẹp

TRÁNH:
❌ Database trống
❌ Phim không có poster (hiện placeholder xấu)
❌ Booking toàn pending
❌ Dashboard revenue = 0
```

3.3. Backup plan khi demo lỗi

Lỗi server không chạy:

- Chuẩn bị video recording demo
- Screenshots các màn hình chính

Lỗi một tính năng:

```
"Tính năng này đang gặp issue, em sẽ show qua code logic.
[Mở file Controller, giải thích flow]
Trong điều kiện bình thường, nó sẽ hoạt động như thế này..."
```

Lỗi database:

- Export file SQL backup
- Có thể import lại trong 30 giây

PHẦN 4: CHEAT SHEET - QUICK REFERENCE

4.1. Các con số cần nhớ

```
KIẾN TRÚC:
- 9 Admin Controllers
- 18+ View files
- 3 loại ghế: Standard, VIP (+50%), Couple (+100%)
- 6 trạng thái booking: pending, confirmed, checked_in, completed, cancelled,
expired

BẢO MẬT:
- 5 lớp: Auth, Role Middleware, CSRF, Eloquent ORM, Blade escaping
```

```
CÔNG NGHỆ:
- PHP 8.x, Laravel 10.x
- MySQL 8.x
- Bootstrap 5
- jQuery (cho interactive)

THUẬT TOÁN:
- SHA-256 cho QR code
- Transaction cho booking
- Eager Loading cho query optimization
```

4.2. Thuật ngữ cần biết

Thuật ngữ	Giải thích đơn giản
MVC	Tách code thành 3 phần: Model (data), View (hiển thị), Controller (xử lý)
Middleware	"Bảo vệ" kiểm tra trước khi cho vào
Eloquent ORM	Thao tác database bằng code PHP thay vì SQL
Migration	"Phiên bản" của database, như Git cho code
CSRF	Chống tấn công giả mạo request
N+1 Problem	Query database quá nhiều lần trong loop
Eager Loading	Load data liên quan cùng lúc để tránh N+1
Transaction	Đảm bảo tất cả hoặc không gì cả
Soft Delete	Đánh dấu xóa thay vì xóa thật

4.3. Flow quan trọng

```
BOOKING FLOW:
User chọn phim → Chọn suất chiếu → Chọn ghế →
Xác nhận giá → Thanh toán → Nhận QR → Check-in

ADMIN CRUD FLOW:
List (index) → Create form → Store →
Edit form → Update → Delete

AUTHENTICATION FLOW:
Login form → Validate credentials →
Create session → Redirect based on role
```

PHẦN 5: BODY LANGUAGE & PRESENTATION

5.1. Khi trình bày

```
☑ NÊN:
- Đứng thẳng, tay để tự nhiên hoặc cầm pointer
- Nhìn vào hội đồng (không nhìn màn hình)
- Nói chậm, rõ ràng
- Dùng tay chỉ khi cần nhấn mạnh

❌ KHÔNG:
- Đọc slide/màn hình
- Nói quá nhanh
- Tay đút túi hoặc khoanh tay
- Nói "ờ", "à", "kiểu như" quá nhiều
```

5.2. Khi trả lời câu hỏi

```
BƯỚC 1: NGHE HẾT CÂU HỎI
- Đừng ngắt lời
- Gật đầu nhẹ để thể hiện đang lắng nghe

BƯỚC 2: DỪNG 2-3 GIÂY
- Thể hiện bạn đang suy nghĩ
- Không trả lời vội

BƯỚC 3: TRẢ LỜI CÓ CẤU TRÚC
- "Câu hỏi rất hay ạ. Em xin trả lời..."
- "Về vấn đề này, em có 3 điểm muốn chia sẻ..."

BƯỚC 4: KẾT THÚC RÕ RÀNG
- "Không biết em trả lời như vậy đã đủ chưa ạ?"
- Hoặc dừng và chờ phản hồi
```

5.3. Khi KHÔNG BIẾT câu trả lời

TUYỆT ĐỐI KHÔNG:

- Nói bừa
- Im lặng quá lâu
- Nói "Em không biết" rồi dừng

HÃY nói:

```
Cách 1: Thành thật + Hướng giải quyết
"Em chưa tìm hiểu sâu về vấn đề này.
Nhưng nếu em gặp trong thực tế, em sẽ research về [keyword]
và tham khảo documentation của Laravel."

Cách 2: Liên hệ với cái mình biết
"Em không chắc về chi tiết kỹ thuật của X,
nhưng em biết nó liên quan đến Y mà em đã implement trong phần Z.
```

```
Em có thể tìm hiểu thêm sau buổi bảo vệ."

Cách 3: Hỏi lại để hiểu rõ hơn
"Em muốn hiểu rõ hơn câu hỏi của thầy/cô.
Thầy/cô đang hỏi về [paraphrase] đúng không ạ?"
```

PHẦN 6: CHECKLIST TRƯỚC NGÀY BẢO VỆ

1 tuần trước

```
□ Test toàn bộ tính năng một lượt
□ Chuẩn bị data demo đẹp
□ Backup database
□ Đọc lại toàn bộ documentation đã viết
□ Luyện demo 2-3 lần, bấm giờ
```

1 ngày trước

```
□ Check laptop: pin, adapter, chuột
□ Check phần mềm: XAMPP chạy OK, browser OK
□ Chuẩn bị file backup (video demo, SQL, screenshots)
□ Ngủ sớm, đủ giấc
□ Chuẩn bị quần áo lịch sự
```

Sáng hôm đó

```
□ Đến sớm 30 phút
□ Test thiết bị phòng bảo vệ (projector, internet)
□ Chạy thử XAMPP, mở sẵn browser
□ Hít thở sâu, relax
□ Uống nước
```

Trong phòng bảo vệ

```
□ Chào hội đồng
□ Giới thiệu tên, đề tài
□ Bắt đầu demo đúng timeline
□ Trả lời câu hỏi bình tĩnh
□ Cảm ơn hội đồng khi kết thúc
```

PHẦN 7: CÂU HỎI THƯỜNG GẶP & ĐÁP ÁN MẪU

Q1: "Giới thiệu tổng quan về đề tài"

```
"CineBook là hệ thống đặt vé xem phim trực tuyến, gồm 2 phần chính:

1. Trang User: Cho phép khách hàng xem phim đang chiếu,
   chọn suất chiếu, đặt ghế và thanh toán online.

2. Admin Panel: Cho phép quản lý rạp quản lý phim,
   phòng chiếu, suất chiếu, và theo dõi doanh thu.

Điểm nổi bật:
- Sơ đồ ghế interactive real-time
- Hệ thống giá động theo loại ghế và khung giờ
- QR code check-in thay vé giấy
- Dashboard analytics cho quản lý

Công nghệ: Laravel, MySQL, Bootstrap, jQuery."
```

Q2: "Tại sao chọn đề tài này?"

```
"Em chọn đề tài này vì 3 lý do:

1. Thực tiễn: Rạp phim là lĩnh vực em quan tâm,
   và booking system có nhiều bài toán kỹ thuật hay.

2. Học hỏi: Đề tài đòi hỏi nhiều kỹ năng:
   - Database design với quan hệ phức tạp
   - Real-time interaction cho chọn ghế
   - Business logic cho tính giá
   - Security cho thanh toán

3. Ứng dụng: Có thể mở rộng thành sản phẩm thực tế
   hoặc làm portfolio khi xin việc."
```

Q3: "Khó khăn gặp phải và cách giải quyết?"

```
"Em gặp 3 khó khăn chính:

1. Race condition khi đặt ghế:
   - Vấn đề: 2 user đặt cùng ghế cùng lúc
   - Giải pháp: Database transaction + unique constraint

2. Thiết kế sơ đồ ghế linh hoạt:
   - Vấn đề: Mỗi phòng có layout khác nhau
   - Giải pháp: Lưu row/column, render động bằng CSS Grid

3. Tính giá vé phức tạp:
   - Vấn đề: Giá phụ thuộc nhiều yếu tố
```

```
    - Giải pháp: Công thức: Base + Seat surcharge + Time surcharge

Mỗi khó khăn đều giúp em học thêm kỹ năng mới."
```

Q4: "So sánh với các hệ thống hiện có?"

```
"So với CGV, Galaxy, Lotte:

Giống:
- Flow đặt vé cơ bản
- Sơ đồ ghế interactive
- QR code check-in

Khác (do scope đồ án):
- Họ có: Payment gateway thật, loyalty program, mobile app
- Em có: Focus vào Admin Panel, analytics, demo-able

Nếu phát triển tiếp, em sẽ thêm:
- Tích hợp VNPay/Momo
- Mobile app với React Native
- Push notification"
```

Q5: "Hướng phát triển trong tương lai?"

```
"Em có roadmap 3 giai đoạn:

Giai đoạn 1 - Hoàn thiện (1-2 tháng):
- Payment gateway integration
- Email notification
- Mobile responsive hoàn chỉnh

Giai đoạn 2 - Mở rộng (3-6 tháng):
- Mobile app
- Loyalty program
- Recommendation system

Giai đoạn 3 - Scale (6-12 tháng):
- Multi-cinema support
- Business Intelligence dashboard
- API cho third-party integration

Em đã document chi tiết trong file DESIGN_THINKING_PART2."
```

PHẦN 8: LỜI KHUYÊN CUỐI

Từ những người đã bảo vệ thành công

```
1. "Đừng học thuộc, hãy HIỂU"
   - Nếu bạn hiểu, bạn có thể giải thích bằng nhiều cách
   - Nếu học thuộc, quên 1 từ là quên cả bài

2. "Hội đồng không phải kẻ thù"
   - Họ muốn bạn đỗ
   - Họ hỏi để bạn có cơ hội thể hiện

3. "Sai không sao, quan trọng là cách xử lý"
   - Demo lỗi? Bình tĩnh giải thích
   - Không biết trả lời? Thành thật và nêu hướng tìm hiểu

4. "Tự tin nhưng không kiêu ngạo"
   - Tự tin: "Em đã implement tính năng này"
   - Kiêu ngạo: "Code của em perfect rồi"

5. "Chuẩn bị kỹ = Bớt lo lắng"
   - Luyện demo nhiều lần
   - Chuẩn bị câu trả lời cho câu hỏi thường gặp
   - Có backup plan cho mọi tình huống
```

Affirmation trước khi vào phòng

```
"Tôi đã làm việc chăm chỉ cho project này.
Tôi hiểu những gì tôi đã code.
Tôi có thể giải thích cho bất kỳ ai.
Tôi đã chuẩn bị kỹ lưỡng.
Tôi sẽ bình tĩnh và tự tin.
Dù kết quả thế nào, tôi đã học được rất nhiều."
```

CHÚC BẠN BẢO VỆ THÀNH CÔNG!

Nhớ rằng: Đây chỉ là một cột mốc. Dù kết quả thế nào, bạn đã xây dựng được một project thực sự và học được vô số kỹ năng quý giá.

"*The expert in anything was once a beginner.*"