HOW HE CAN IMPROVE ALL HIS WORK (DETAILED)

1️⃣ Task 1A – The Proposal (Biggest Gains Area)

◇ A. Strengthen Risk Mitigation (Not Just Risk Identification)

Current state

- Risks are identified (data breach, downtime, incorrect information)
- Regulations are named (GDPR, WCAG)

How to improve
For **every risk**, add:

1. The **cause**
2. The **impact**
3. The **mitigation**

Example upgrade

Risk: Data breach of visitor information
Mitigation:

- Enforce role-based access control for staff
- Encrypt personal data at rest and in transit
- Apply data minimisation in line with GDPR principles

👉 This directly pushes **Decomposition** and **Wider Issues** into Band 3.

◇ B. Explicitly Link Regulations to System Features

Current state

- GDPR and WCAG are mentioned correctly

How to improve
Explain **how the system enforces them**.

Example

GDPR compliance is supported through explicit consent prompts, defined data retention periods, and the ability for users to request data deletion.

WCAG compliance is achieved through contrast-tested colour schemes, screen-reader-friendly navigation, and keyboard-only interaction support.

👉 This moves reasoning from *descriptive* → *justified*.

◇ C. Make Decomposition More "Software-Engineering Like"

Current state

- System components are listed (good)

How to improve
Add **phases** or **dependencies**.

Example

The system would be developed in stages, beginning with user account management and navigation services, followed by real-time queue monitoring and analytics once core functionality is stable.

👉 This shows **professional planning**, not just ideas.

2️⃣ Task 1B – Design (Already Excellent, Now Make It Examiner-Proof)

◇ A. Add Accessibility Strategy (Not Just WCAG Testing)

Current state

- WCAG contrast testing shown (very good)

How to improve
Add a **short accessibility strategy section**:

- Keyboard navigation
- Focus indicators
- Alt text policy
- Scalable text

Example

All interactive elements include visible focus states to support keyboard users, and non-decorative images are accompanied by descriptive alternative text.

👉 This locks in **top-band interface design**.

◇ B. Explain Algorithm Efficiency (Tiny Change, Big Impact)

Current state

- Algorithms are correct and clear

How to improve
Add **1–2 justification sentences** per flowchart.

Example

Availability checks are performed before authentication to reduce unnecessary database queries and improve system efficiency during peak usage.

👉 This shows **computational thinking**, not just structure.

◇ C. Strengthen Data Design with Integrity Controls

Current state

- ERD and data dictionary are correct

How to improve
Add a short section:
“Data Integrity and Validation”

Mention:

- Foreign key enforcement
- Preventing orphan records
- Server-side validation

👉 This ties **data → security → reliability** together.

3️⃣ Testing – Make Traceability Crystal Clear

Current state

- Very strong test strategy and coverage

How to improve
Add **requirement references** to tests.

Example

These tests validate the functional requirement that users must be able to book tickets without system errors.

👉 Examiners LOVE traceability.

4️⃣ Documentation – Add Reflection (Maturity Marker)

Current state

- Clear and professional documentation

How to improve
End Task 1B with a **short reflective paragraph**:

- What design choice was hardest?
- What would be improved next?

Example

Given additional development time, automated testing would be introduced to improve reliability during peak visitor periods.

👉 This prepares perfectly for **Task 2 & Task 3**.

🧠 GLOBAL IMPROVEMENT RULE (Very Important)

Across **all tasks**, he should:

- Replace “The system will…” with
  “This approach was chosen because…”
- Always link:
  feature → user need → business benefit