**📑 Distinction-Level Task 1 – Model Answer (GibJohn Tutoring)**

**Part A – Analysis**

**1. Introduction**

The purpose of this document is to analyse the needs of GibJohn Tutoring and propose a digital solution that addresses their requirements. This includes exploring existing platforms, identifying functional and non-functional requirements, and ensuring legal, regulatory, and professional compliance.

**Client**: GibJohn Tutoring provides face-to-face tutoring and learning resources.  
**Objectives**: Create a digital platform that supports interactive resources, wider learning, and progress monitoring.  
**Link to business goals**: Extend reach, improve learner engagement, and futureproof services with emerging technologies.

**2. Analysis of Existing Solutions**

**Strengths**:

* *Khan Academy* → strong content variety and assessments.
* *Duolingo* → effective gamification.

**Weaknesses**:

* Limited accessibility options.
* Lack of deep personalisation.

**Opportunities**:

* Introduce AI-driven personalisation.
* Accessibility-first design.
* Gamified rewards tied to learning outcomes.

**Justification**: A new solution is needed to combine the strengths of existing platforms with accessibility, compliance, and tailored learning for GibJohn’s audience.

**3. Emerging Technologies**

* **AI**: Adaptive quizzes, automated marking.
* **IoT**: Smart classroom devices linking to the platform.
* **AR/VR**: Virtual lab experiments and immersive subject exploration.
* **Cloud computing**: Scalable, cost-efficient hosting.

**Integration**: AI and Cloud included immediately; IoT and AR/VR reserved for future iterations.

**4. Business Context**

**Overview**: Tutoring company expanding digitally.  
**Goals**: Wider access, improved engagement, measurable learner progress.  
**Audience**: Learners (school-aged), tutors, guardians.  
**Constraints**: Budget, compliance with GDPR/Equality Act, phased implementation.

**5. Requirements Analysis**

**Functional Requirements**

* Learner registration and login.
* Tutor uploads and resource sharing.
* Progress tracking dashboards.
* Reward system for achievements.

**Non-Functional Requirements**

* Secure (password hashing, HTTPS).
* Accessible (WCAG 2.1).
* Reliable (99.9% uptime).
* Usable (simple navigation, responsive).

**KPIs**

* < 2s page load time.
* 99.9% uptime.
* 80% weekly active learner usage.
* 90% satisfaction in UAT.

**MoSCoW**

* Must: Login, progress tracking, content delivery.
* Should: Rewards and accessibility features.
* Could: AI-driven personalisation.
* Won’t: VR integration at launch.

**6. Problem Decomposition**

**Modules**:

* Authentication.
* Resource Management.
* Progress Tracking.
* Reward System.
* Admin Dashboard.

**System Architecture**:

Frontend (React) → Backend (Django) → Database (MySQL)

↘ AI Analytics Engine ↙

Inputs: user credentials, content, progress data.  
Processes: validation, storage, tracking, analysis.  
Outputs: dashboards, reports, rewards.

**7. Proposed Solution**

* **Stack**: React (frontend), Django (backend), MySQL (database).
* **Hosting**: AWS Cloud.
* **Security**: Encrypted logins, HTTPS, role-based access.
* **Emerging Tech**: AI for adaptive learning, Cloud for scalability.

Justification: Secure, scalable, industry-standard technologies aligned to GibJohn’s needs.

**8. Legal & Regulatory Compliance**

| **Law / Standard** | **Requirement** | **How Addressed** |
| --- | --- | --- |
| GDPR | Protect learner data | Encryption, consent, anonymisation |
| Equality Act 2010 | Accessibility | WCAG 2.1 compliance, screen reader support |
| Consumer Rights Act | Transparent, fair terms | Clear T&Cs, opt-out options |
| Copyright/IP law | Respect ownership | Licensed resources, attribution |
| W3C Standards | Accessible web design | Semantic HTML, ARIA roles |

**9. Professional and Ethical Standards**

Aligned with **BCS Code of Conduct**:

* *Public interest*: Accessibility-first design.
* *Integrity*: Transparent handling of learner data.
* *Competence*: Secure, reliable development practices.
* *Privacy*: GDPR-compliant data storage.

**10. Justification of Design Choices**

* **React**: Responsive, fast, widely supported frontend.
* **Django**: Secure backend with authentication.
* **MySQL**: Relational database suitable for structured learner/tutor data.
* **AWS**: Scalability and uptime guarantees.
* **UI/UX**: Chosen for accessibility, readability, and cross-device use.

**11. Risk Assessment**

| **Risk** | **Impact** | **Likelihood** | **Mitigation** |
| --- | --- | --- | --- |
| Data breach | High | Medium | Encryption, 2FA |
| Missed deadlines | Medium | High | Agile sprints, weekly reviews |
| System downtime | High | Low | Cloud redundancy, monitoring |
| Low learner uptake | Medium | Medium | Gamification, user testing |
| Copyright infringement | High | Medium | Use licensed materials only |

**Part B – Design**

**12. Interface Design**

**Wireframes**:

* Login Page (credentials).
* Dashboard (progress bars, resources, rewards).
* Tutor Portal (upload + learner tracking).

**Navigation Flow**: Login → Dashboard → Content → Assessment → Rewards.  
**Justification**: WCAG 2.1 compliance, clear layout, responsive for mobile/tablet.

**13. Data Design**

**ERD**:

* Entities: User, Tutor, Resource, Progress, Reward.
* Relationships: One user → many progress records, one tutor → many resources.

**Data Dictionary Example**:

| **Entity** | **Attribute** | **Type** | **Example** |
| --- | --- | --- | --- |
| User | userID | INT PK | 101 |
| User | email | VARCHAR | user@mail.com |
| Progress | score | INT | 85 |
| Reward | points | INT | 150 |

Error handling: input validation, length limits, clear error messages.

**14. Algorithm Design**

**1. Registration**

Input: username, password

Hash password

Store in database

**2. Login**

Input credentials

Hash + compare

If match → dashboard

Else → error

**3. CRUD (Resource Upload)**

If tutor role verified

Accept file

Insert into DB

**4. Reward System**

If learner completes activity

Add points to profile

Check milestones → award badge

**5. Feedback Submission**

Input text

Validate + store in DB

Notify tutor

**15. Test Strategy**

**Test Levels**:

* Unit → login, resource upload.
* Integration → login + DB → dashboard.
* UAT → solution vs KPIs with end users.

**Test Data**: normal, erroneous, extreme (boundary) inputs.

**Sample Test Cases**:

| **Test Case** | **Input** | **Expected Output** |
| --- | --- | --- |
| Login valid | Correct creds | Dashboard loads |
| Login invalid | Wrong pass | Error message |
| Add resource | Upload PDF | Resource visible |
| Progress update | Complete quiz | Dashboard shows new score |

**Order**: Unit → Integration → UAT.

**Distinction-Level Extras**

✔ Compliance mapping table (Legal & Regulatory).  
✔ Diagrams (wireframes, ERD, flowcharts).  
✔ Explicit BCS Code of Conduct reference.  
✔ Emerging tech (AI now, AR/VR later).  
✔ Justification of all design decisions.  
✔ Structured, professional communication for technical + non-technical readers.