Student ID No: \_\_\_\_\_\_\_\_\_\_Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_

Course & Year/Section: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Score: \_\_\_\_\_\_\_\_\_\_\_\_

**PERFORMANCE TASK 4**

**CONTEXT DIAGRAM, PARENT DFD & HIGH-LEVEL SYSTEM OVERVIEW**

**General Directions**

* Read and understand each statement carefully.
* Write your answers on your activity sheet.

**Objective:**

By the end of this lab, students should be able to:

1. Develop a system context diagram showing how the proposed system interacts with external entities.
2. Create a parent diagram (Level 0 DFD) to represent the major processes and data flows within the system.
3. Write a high-level system overview describing system components and their roles.
4. Apply principles of abstraction and modularity in system design.
5. Collaborate effectively using version control tools.

**Step 1: Review Week 1 Deliverables**

* Open your team’s repository.
* Review your ProjectOverview.md — ensure the objectives and scope are clearly defined.
* Discuss any updates or refinements needed based on feedback.

**Step 2: Create a Context Diagram**

1. Identify 3–5 external entities (e.g., users, external systems, databases, APIs).
2. Represent your proposed system as a single process box.
3. Show data flows between external entities and the system.
   * Example: User → Login Request → System → Authentication Response → User
4. Use clear labels for inputs and outputs.

Save as: ContextDiagram.png or ContextDiagram.pdf in the /docs folder.

**Step 3: Create the Parent Diagram (Level 0 DFD)**

1. Break down the system into 3–5 major processes (e.g., Authenticate User, Process Payment, Generate Report).
2. Identify data stores and external entities interacting with these processes.
3. Show data flows between:
   * Processes ↔ Data Stores
   * Processes ↔ External Entities
   * Processes ↔ Other Processes

Save as: ParentDFD.png or ParentDFD.pdf in the /docs folder.

**Step 4: High-Level System Overview**

In your ProjectOverview.md, add a new section titled “High-Level System Overview”. Include:

1. Major Modules/Subsystems – List and briefly describe at least 3 modules.
2. External Systems/Interfaces – Mention APIs, third-party services, or databases.
3. Data Flow Summary – A short narrative explaining how data moves across the system.

**Step 5: Version Control & Collaboration**

1. Assign one member to upload the diagrams.
2. Another member should update the documentation.
3. All members must review and approve changes via pull requests.

**Deliverables**

1. ContextDiagram.png/.pdf in /docs
2. ParentDFD.png/.pdf in /docs
3. Updated ProjectOverview.md with High-Level System Overview
4. Git repository updated with contributions from multiple members

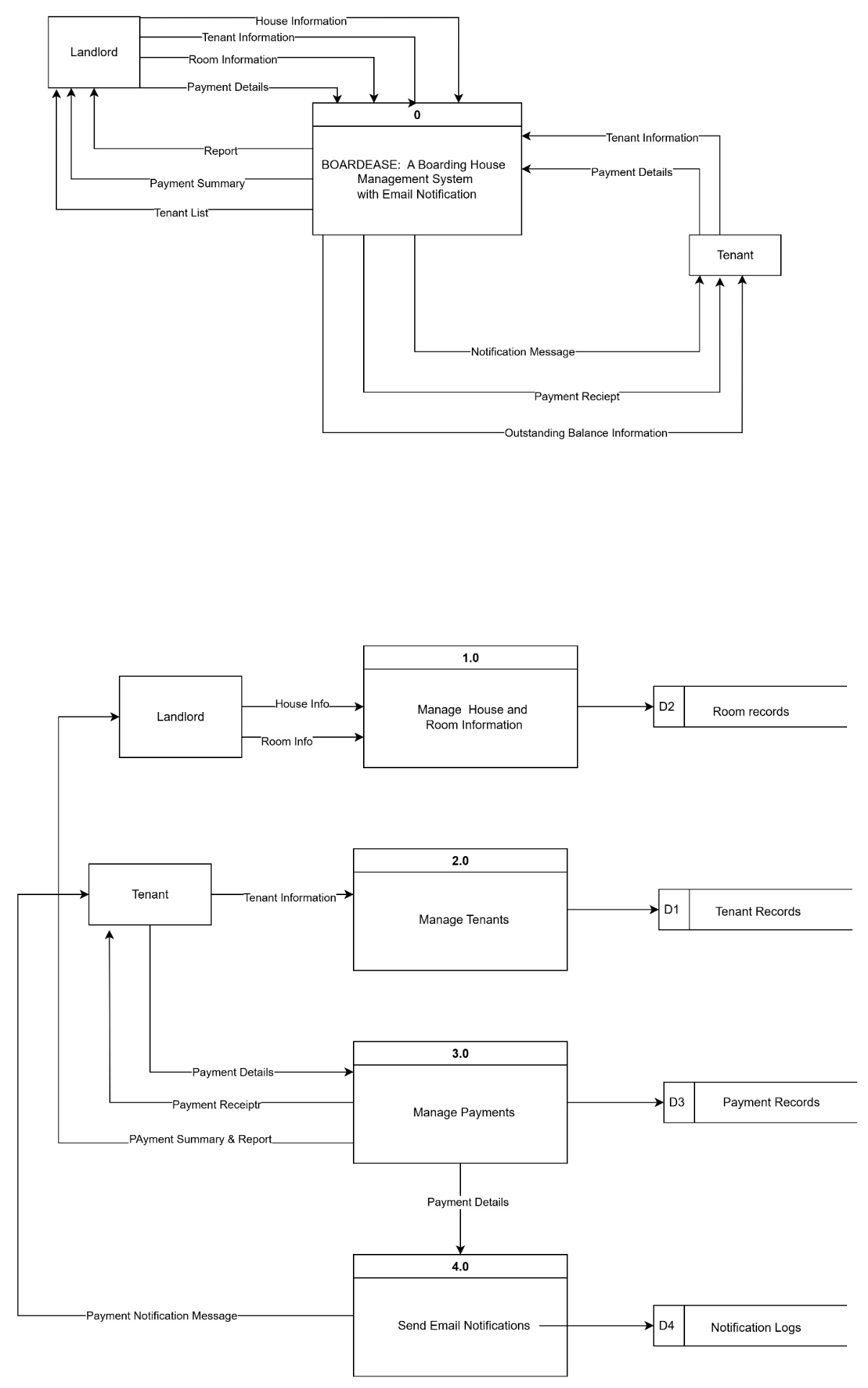
**Evaluation Criteria (Total: 100 pts)**

| **Criteria** | **Points** |
| --- | --- |
| **Context Diagram** – Clarity, correctness of entities and flows | 30 pts |
| **Parent DFD** – Logical structure, correct use of processes/data stores | 30 pts |
| **Documentation** – High-Level Overview section is clear and aligned | 20 pts |
| **Repository Update** – Proper organization and version control usage | 10 pts |
| **Team Collaboration** – Evidence of multiple members contributing | 10 pts |

A screenshot of a computer

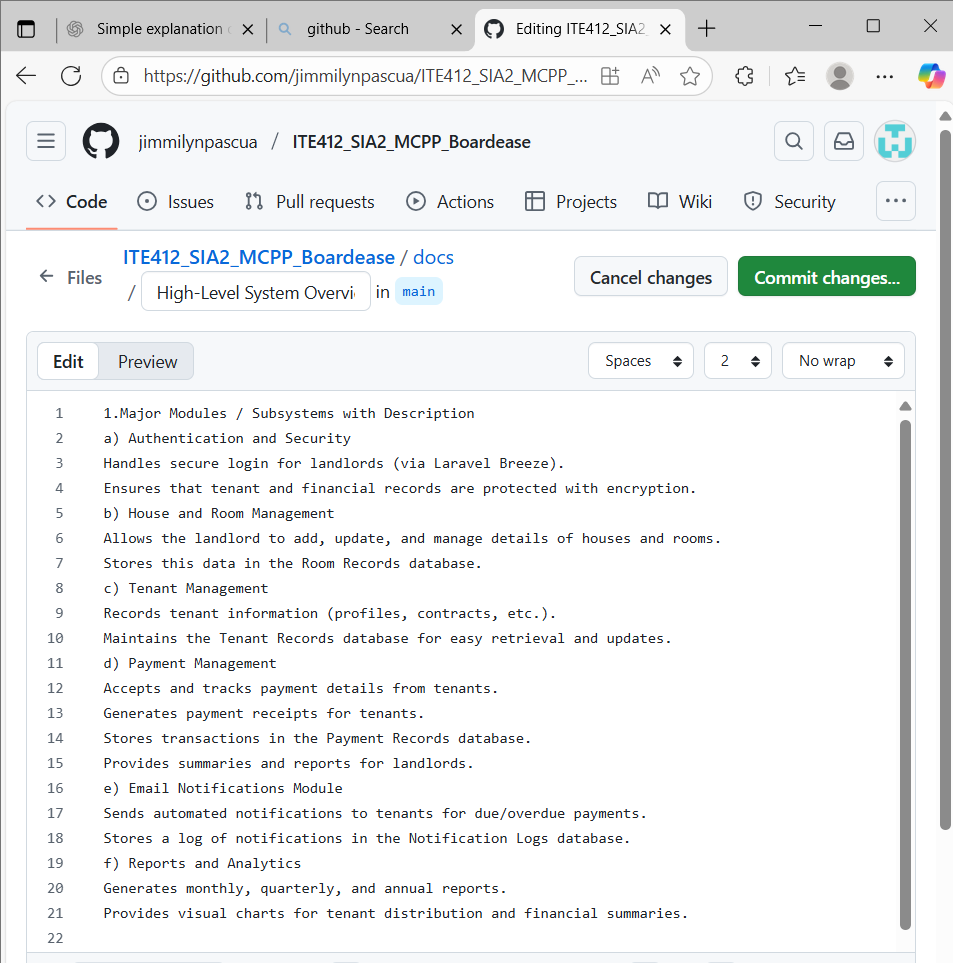
AI-generated content may be incorrect.

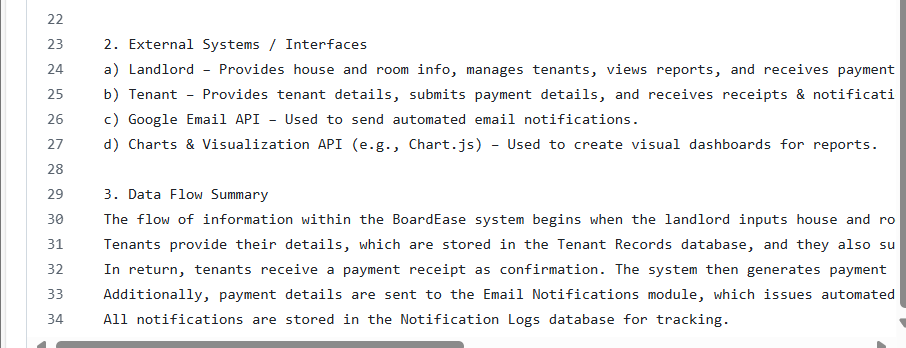
**CONTEXT DIAGRAM**



**PARENT DIAGRAM**

**HIGH-LEVEL SYSTEM OVERVIEW**





The flow of information within the BoardEase system begins when the landlord inputs house and room information, which is stored in the Room Records database.Tenants provide their details, which are stored in the Tenant Records database, and they also submit payment information, which is processed and saved in the Payment Records database.In return, tenants receive a payment receipt as confirmation. The system then generates payment summaries and detailed financial reports for landlords, covering monthly, quarterly, and annual periods.Additionally, payment details are sent to the Email Notifications module, which issues automated reminders to tenants about due or overdue balances via the Google Email API. All notifications are stored in the Notification Logs database for tracking.