A blue circle with white text

Description automatically generated

**COMSATS UNIVERSITY ISLAMABAD, ABBOTTABAD**

**Software Design and Architecture**

**Assignment - 01**

***Submitted by:***

Laiba Binta Tahir FA21-BSE-019

Arfah Ali FA21-BSE-080

***Submitted to:***

Mam Neeli khan

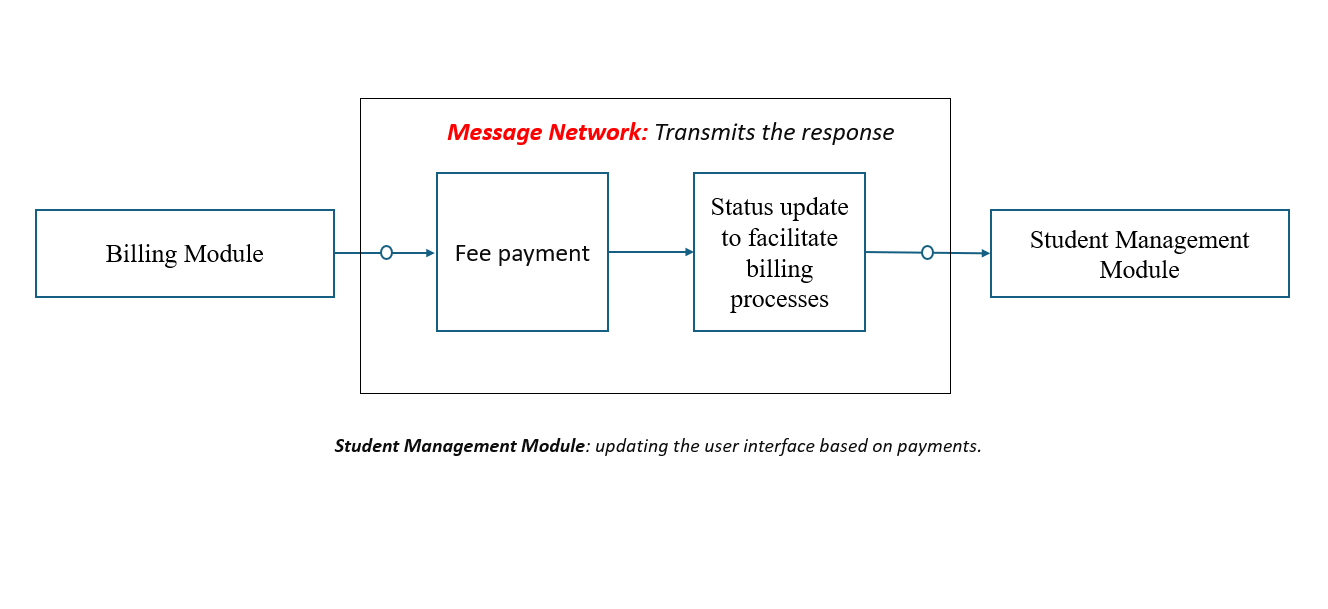
**High-Level S/w Architecture for Hostel Management System**

**Major Elements:**

1. User Interface Module: This module handles user interactions, including students, staff, and administrators. It provides interfaces for tasks such as room booking, check-in/check-out, fee payment, and complaints.
2. Student Management Module: Manages student information such as registration details, room allocation, and preferences. It also handles requests for room changes and provides functionality for managing student accounts.
3. Room Management Module: This module oversees room allocation, availability, maintenance scheduling, and inventory management for hostel facilities.
4. Staff Management Module: Manages staff information including roles, duties, shifts, and payroll.
5. Billing and Finance Module: Handles fee collection, billing, invoicing, and financial reporting for students and hostel administration.
6. Complaints and Maintenance Module: Allows students to submit complaints or maintenance requests and facilitates the tracking and resolution of these issues by maintenance staff.
7. Reporting and Analytics Module: Provides administrators with insights through reporting and analytics functionalities, such as occupancy rates, revenue analysis, and student feedback analysis.

**Connectors:**

1. User Interface Module to Student Management Module:
   * Type: Two-initiator connector
   * Justification: This connector allows bidirectional communication between the user interface and the student management module. The user interface may initiate requests, such as submitting registration details or querying room availability. Similarly, the student management module may also trigger actions or updates in response to user interactions, like confirming room assignments or updating student information.
2. Student Management Module to Room Management Module:
   * Type: One-initiator connector
   * Justification: In this connector, the student management module initiates, meaning it can send requests or initiate actions to the room management module. Students may request room changes, check room availability, or submit preferences through the user interface, triggering actions in the room management module. However, the room management module doesn't initiate requests to the student management module because it primarily responds to student-initiated actions.
3. Room Management Module to Staff Management Module:
   * Type: One-initiator connector
   * A diagram of a network

     Description automatically generatedJustification: Here, the room management module initiates communication with the staff management module. This allows the room management module to notify staff about maintenance schedules, request assistance for resolving issues, or allocate tasks to staff members. However, the staff management module doesn't initiate requests to the room management module because it primarily responds to room-related events or requests initiated elsewhere in the system.
4. Billing and Finance Module to Student Management Module:
   * Type: Two-initiator connector
   * Justification: This connector enables both modules to initiate communication as needed. The billing and finance module may need to confirm fee payments, update student accounts, or generate invoices, which can trigger actions in the student management module. Similarly, the student management module may need to provide student information or status updates to facilitate billing processes, requiring communication with the billing and finance module
5. Complaints and Maintenance Module to Staff Management Module:
   * Type: One-initiator connector
   * Justification: The complaints and maintenance module initiates requests to the staff management module, allowing maintenance staff to update the status of complaints, request additional resources, or assign tasks to staff members. However, the staff management module doesn't initiate requests to the complaints module because it primarily responds to maintenance-related events or requests initiated elsewhere in the system.
6. Reporting Module to Student Management Module and Billing and Finance Module:

* Type: One-initiator connector
* Justification: This module pulls data from the student module and finance module to create reports. This module only retrieves the necessary information for reporting and analytics such as student info, payment method, etc. This doesn't initiate requests to the student module.

1. Reporting Module to User Interface Module:

* Type: One-initiator connector
* Justification: This connector allows the reporting module to provide summarized data or visuals to the user interface for wardens to view. The reporting module initiates communication to deliver relevant data or reports to the user interface, enhancing the user experience and helping the administrators or wardens in decision-making.

1. SQL queries:

* Type: Data access connector
* A diagram of a data flow

  Description automatically generatedJustification: This connector allows the DAL (Data Access Layer) to interact with databases using queries of SQL. The DAL built & executes the queries to manipulate data in DB. These queries provide a systematic way to communicate with the HMS database to perform operations like retrieving records of students, updating room status, and report generation. Thus, this connector helps in data access and manipulation within HMS.

1. Request/response connector:

* Type: synchronous connector
* Justification: This connector guarantees synchronous communication b/w user interface and business logic layer. When a user interacts with an interface such as checking room availability or giving feedback, the user sends a request to the Business logic layer; which processes the request and sends back the response to the UI and updates the interfaces accordingly.

1. API calls

* Type: synchronous connector
* Justification: This connector ensures synchronous communication b/w the Business logic layer and the Data Access Layer. When BLL wants to perform any CRUD operation on the database, it makes an API call to the Access Layer. These calls allow Business logic layer to perform operations from the database as required.