

~~say~~ Develop a complete IEEE standard SRS document with several equipments

HOTEL MANAGEMENT SYSTEM

1. Introduction :-

(a) Purpose :-

This document outlines the functional and non-functional requirements for the Hotel Management System. It will serve as a guide for developers, testers and stakeholders to understand the expected features and performance of the system. The goal is to automate hotel operations and improve customer service.

(b) Document Conventions :-

All headings are bold and properly numbered for easy reference.

Requirements are labeled as Functional, Non-functional, External Interface.

Standard English is used throughout for clarity.

(c) Intended Audience and Reading Suggestions

The SRS document intended for:

Developers, testers, project managers and hotel administrators. Developers should focus on requirements, testers on

functionality and interfaces while managers

and clients may focus on scope and user roles.

(a) Project Scope & Functionality

The HMC will manage hotel operations such as reservations, room availability, customer check-in, check-out, staff billing.

(c) References :-

- IEEE 830 Software Requirements Specification Standard.
- MySQL and Java coding documentation.
- API's for online payment and mail notification integration or notifications.

2. Overall Description

① Product Perspective :-

The HMC is a standalone application but may be extended to integrate with payment gateways and email services. It is intended for both local use and online access.

② Product functions :-

Main functions include room reservations, check-in/check-out, billing, report generation and housekeeping status tracking. It also allows admin-level control for staff management.

- (c) User Classes and Characteristics :-
- Administrator :- full access to all modules and data management.
 - Receptionist :- can manage bookings and billings.
 - Housekeeping staff :- can view and update room cleaning status.
 - Customer :- can book rooms, culture answer ticket books, view history.

- (d) Operating Environment :-
- The system will run on windows.
It uses MySQL for backend and Java for frontend. Web components should be compatible with modern browsers.

- (e) Design and Implementation constraints :-
- It must be scalable, secure and support integration with external services.
The system should use only open source technologies.

- (f) User Documentation :-
- User guides and admin guides will be provided. These will include system navigation, features, FAQ, and troubleshooting.

- (g) Assumptions and Dependencies:-
- Assumes users have basic computer knowledge. Online features depend on internet availability. The system also depends on correct setup of hosting environment.

(3) Specific Requirements :-

a) Functional Requirements :-

- Users can book, modify and cancel reservations.
- Receptionist manages check-ins and check-outs.
- System generates and prints bill.
- Admin can add/edit rooms and view reports.
- Email confirms bills sent after booking.

b) Non-functional Requirements :-

- System should respond within 2 seconds.
- 99.9% uptime is expected.
- Secure login and encrypted data storage.
- Must support at least 20 users at the same time.
- Should be user friendly and fast.

c) External Interface Requirements :-

- GUI for admin, staff and customers.
- Integration with payment gateways.
- Email server for sending notifications.
- optional barcode scanner support at reception.

d) Appendices :-

- Necessary terms used in HMS.
- Screenshot mockups and interfaces.
- Use Case, ER and AFD diagrams.
- sample test case for each page.

2 CREDIT CARD PROCESSING

1. Introduction :-

(a) Purpose :- This system securely processes credit card payments, including validation, authorization, fraud check and transaction recording. It ensures fast, reliable and compliant payment handling.

(b) Document Conventions :-

Follows IEEE 880 format using clear section numbering and labels linking functional requirements, non-functional and external interface. Language is simple and consistent.

(c) Intended Audience :-

For developers, testers, security analysts each IT should focus on relevant sections like requirement and performance criteria.

(d) Project Scope :-

The system handles online and offline credit card ~~service~~ transactions, detects and validate card details.

(e) References :-

- IEEE 880
- PCI DSS guidelines
- VISA and Mastercard API docs
- ISO smart card Standard

2. Overall Description

(a) Product Perspective: ~~Architecture~~
Acts as middleware between merchants, banks and card networks. Integrates with third-party payment gateways for secure transactions.

(b) Product Functions: ~~Architecture~~
Includes performance validation, payment approval, fraud detection, transaction logging and reporting.

(c) User Classes: ~~Architecture~~
Merchant initiates and monitors transactions
Cardholder makes payment
Admin manages users and logs
Security Team: Monitors and sends alerts

(d) Operating Environment: ~~Architecture~~
Runs on a secure web server using HTTPS, supporting web and mobile POS systems and using encrypted connections.

(e) Implementation Constraints: ~~Architecture~~
Can't store sensitive card data without compliance. Must integrate with API's from banks and card networks and ensure high performance.

(f) User Documentation: ~~Architecture~~
Includes guides for merchants, admin panels, API integration and security operations.

(g) Assumptions:
Requires internet, verified users and third party fraud detection systems for smooth operations.

3. Specific Requirements

- @ Functional Requirements:
- Validate card details and approve/reject payments
 - Record transactions and notify users
 - Block suspicious activity
 - Provide summary reports

⑥ Non-functional Requirements:

Response time less than 3 seconds.
System should be available 24/7
with 99.9% uptime.

End-to-end encrypted data transmission
Scalable to handle peak load traffic

⑦ External Interface Requirements:-

Integration with banking gateway APIs
GUI dashboards for merchants and admins
Communication via secure sockets
Support web and mobile interface.

4. Appendices:

Glossary, use case and sequence diagrams for transactions, API request format examples, sample test case and validation criteria.

3 STOCK MAINTENANCE SYSTEM

Business requirement, Functional requirement

1. Introduction to Stock Maintenance System

(a) Purpose: - To manage and track stock digitally with real time updates.

(b) Conventions in "Shallow" Business Form:

Business requirements have started changing

(c) Intended audience of business

Managers, Developers, Testers, Staff

Overall Description:-

- At beginning of project I have written

RFID perspective in which we integrated

which has standalone of future integration

with POS system in future

function window has been triggered

(d) Functions:- Add / Delete / update items, stock alerts, reports, search/filter

update role based on access, product

trading through IGA & maintenance of

(e) User Roles Admin (full control), Staff (update / report), Viewer, visitor

- ① operating environments: Windows / Linux,
MySQL, DB, Desktop
- ② Design and implementation constraints:
Relational DB, security policies,
LAN/WAN access.
- ③ User Documentation: User manual,
online help.
- ④ Assumptions and Dependencies: Users are
computer-literate, data entered
correctly.

3. Specific Requirements:-

- ① Functional Requirements:-
The system manages stock by adding,
updating, deleting items, showing
real time availability, generating
low-stock alerts, reports and keeping
transaction log with role based access.
- ② Non functional requirements:-
It ensures fast response and
scalability upto 10,000 items, security
with encryption, user friendly design,
high uptime and easy maintenance.
- ③ External Interface Requirements:-
The system provides API with dash-
boards, connects to database like SQL
works on standard OS, communicates

via LAN/WAN and supports future
POS integration. 84, 197. vi

(d)

Appendices: Business plan, budget

Glossary, stock, report, threshold
future enhancements, Mobile app,
barcode scanner, auto supplier

Item ordering, vendor selection, cost
plan, budget

Small advantages over alternatives
fewer items stored, integrated
functions

- streamlined workflow

streamlined, lowest cost

fastest and least expensive methods and
processes, more efficient, profitable
process, validation and low
processing time, strong, highly automated
cost benefit, fast return on investment

streamlined, lowest cost

fastest, most accurate, less expensive
process, validation and cost reduction
environment, fast, high efficiency, high
accuracy

streamlined and cost effective

4. PASSPORT AUTOMATION SYSTEM

1. Introduction

- ① Purpose : the system automates passport application, verification and issuance, ultimately reducing delay and manual work.
- ② Document conventions : All requirement would use "shall", categories are functional, non-functional and interface.
- ③ Intended audience and reading suggestion : applicants, govt officials, developers, tester and managers.
- ④ Project scope : the system enables online applications, payment, tracking, verification and final passport delivery.
- ⑤ References : IEEE std 830, official govt e-governance policies and existing passport guidelines.

2. Overall Description

- ① Product Perspective : a web based solution replacing manual process designed for integration with govt databases.

⑥

Product Functions: Users apply online
upload documents, pay fees, schedule
appointments, track status, etc.
officials issue passports

⑦

User classes and characteristics:

Applicants submit requests, verifies
check and approve document
and admins control the system

⑧

Operating Environment: works on
windows / LINUX servers, underlying
MySQL DB, accessible via common
browsers

⑨

Design and implementation constraints:
Must follow good data protection,
security standards and handle
large user traffic

⑩

User Documentation: Provides user
manuals, online help and FAQ's for
users

⑪

Assumptions and Dependencies: users
have internet access and valid documents
system depends on secure payment
methods

↳ Assumptions: users aged 18+

↳ Assumptions: users aged 18+

↳ Assumptions: users aged 18+

⑥ Specific Requirements

① Functional Requirements :- Applicants can register, apply, upload, pay, track applications while officials qualify, update and issue passports.

② Non-functional Requirements

The system shall be fast, secure, reliable, scalable and easy to use.

③ External Interface Requirements

Provides browser based GUI, connects to databases, integrates with payment verification system over internet / LAN.

Appendices :-

Glossary :-
Applicant = citizen user,
Verifier = govt officer, Status = Current
Stage

Future Enhancements :- integration with mobile app, biometric verification and fraud detection

5. LIBRARY MANAGEMENT SYSTEM

- ① Introduction: It is a management system used for managing library resources like books, journals, magazines, etc. It helps in automating various library operations such as book issue, return, cataloging, and member management to save time and reduce errors.
- ② Purpose: The purpose of the system is to automate library operations including book issue, return, cataloging, and member management to save time and reduce errors.
- ③ Document Convention: All req. should use "shall", categorize statements into functional (functional) and non-functional (non-functional), and external interfaces in terms of data formats, protocols, and message exchange mechanisms.
- ④ Intended audience and Reading suggestion: Librarians, students, faculty, developers, testers and managers.
- ⑤ Project Scope: The system manages book records, member accounts, transactions and generate reports.
- ⑥ References: IEEE std 830, library automation standard, and institutional library guidelines.

2. Overall Description :-

- ① Product Perspective :- A stand-alone solution replacing manual registers with options for future integration.

- (b) Product function: allows adding, removing books, issuing, returning, generating reports and provide search functionality.
- (c) User classes and characteristics:-
librarian manages books and members
students borrow book and administers the system.
- (d) operating environment:- works on various LINUX with MySQL DB and accessible via web browsers.
- (e) Design implementation constraints:-
must ensure data consistency, prevent issues of same copy and follow institutional policies.
- (f) User Documentation:- includes manuals for staff, FAQ's for students and training guides.
- (g) Assumptions and Dependencies:-
User should have valid accounts, system relies depends on reliable database and network connection.
3. Specific Requirements:-

- (1) Functional Requirements:- This system should allow book issue / return, manage due dates and fines.

plan and generate usage reports.

- (B) Non-functional Requirements:
- It shall provide responses within 2 sec to ensure security.
 - Maintain 99% uptime and user friendly interface.
- (C) External Interface Requirements:
- Provides a GUI with search and menus, connects to relational databases and communicates over LAN / Internet of devices.

Appendices: -
Glossary:
Member = student / faculty
librarian = staff, transaction = issue / return
future enhancement = barcode support,
e-book access, mobile app,

auto fine payment

cooperative loans worth mentioning

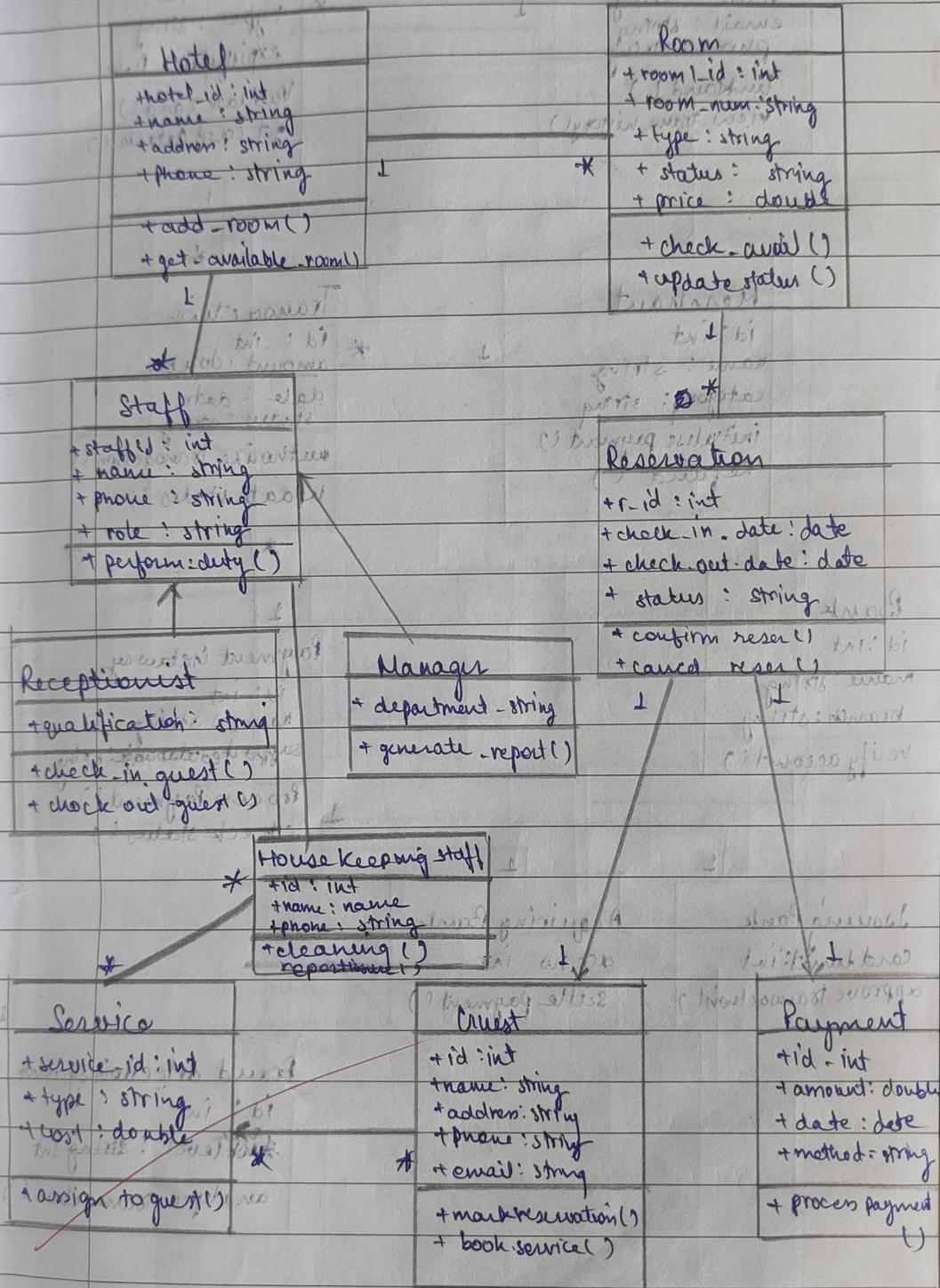
student bills and library's recall
will be done automatically with a notice
with whom it is due and details

- international signs

and anti-storming features

Class Diagrams

I HOTEL MANAGEMENT SYSTEM

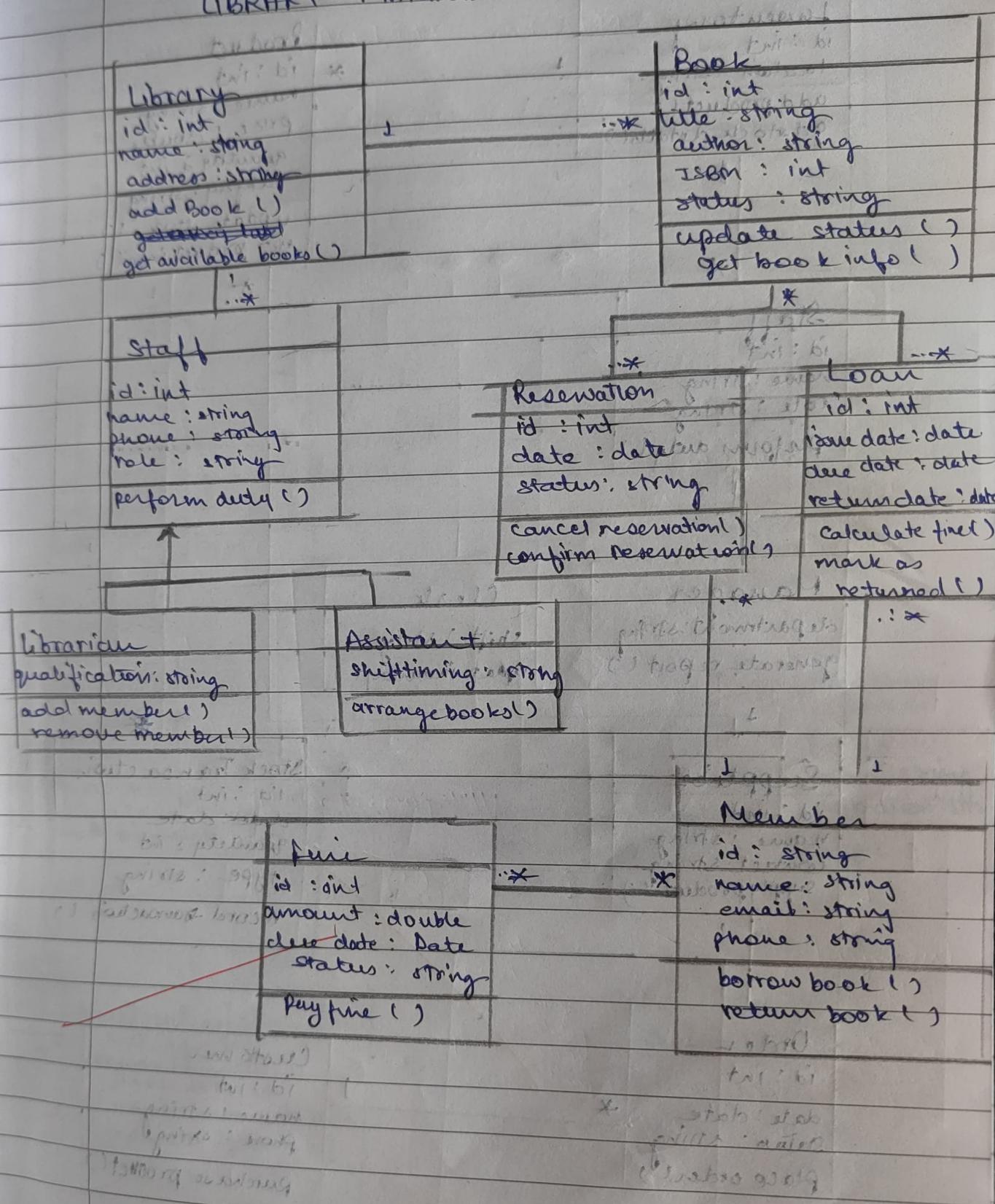


CREDIT CARD PROCESSING

Customer	id : int name : string email : string phone : string password : string linkand() view_trans_history() create()	1	* number : int * type : string * expiringdate : Date ValidateCard() get_card_details()
Merchant	id : int name : string category : string initialise_payment() Refund() view_bills()	1	* id : int amount : double date : date status : string authorize_transaction() update_status()
Bank	id : int name : string branch : string verify_account()	1	* id : int name : string supported_database : string process_payment() check_status()
Issuing Bank	card_limit : int approve_transaction() block() debit : float credit : float freeze : float unfreeze : float	1	* id : int acc_no : int settle_payment()
Acquiring Bank	1	1	* id : int acc_no : int process : float process_error : float process_time : float process_type : string process_update : float process_update_error : float process_update_time : float process_update_type : string process_update_value : float process_update_warning : float process_update_warning_error : float process_update_warning_time : float process_update_warning_type : string
Fraud Detection System			* id : int risk_level : string analyze_transaction()

STOCK MAINTANANCE SYSTEM

LIBRARY MANAGEMENT SYSTEM



STOCK MANAGEMENT SYSTEM REPORT 2.

Date _____
Page _____

```
Inventory  
id : int  
location : string  
add product()  
get stock details()
```

```
* Product
    id : int
    name : string
    price : double
    quantity : int
    Category : string
    updateStock(12)
    getProdInfo()
```

	<u>Staff</u>	
X--	id : int	X--
never	name : string	never
tri : int	role : string	tri : bit
stab : stab	perform duty()	stab
stab : stab	get role : int	stab
stab : stab	↑	stab
stab : stab name : string	(Incubator or Listener)	
(Leaf staffs)	Main function of manager:	
do work		
(1) Inheritance	<u>Manager</u>	Cl
X--	department : string	
	generate report()	shift
		update
		Customer

Clerk

Supplier
id : int
name : string
contact : string
supply product ()

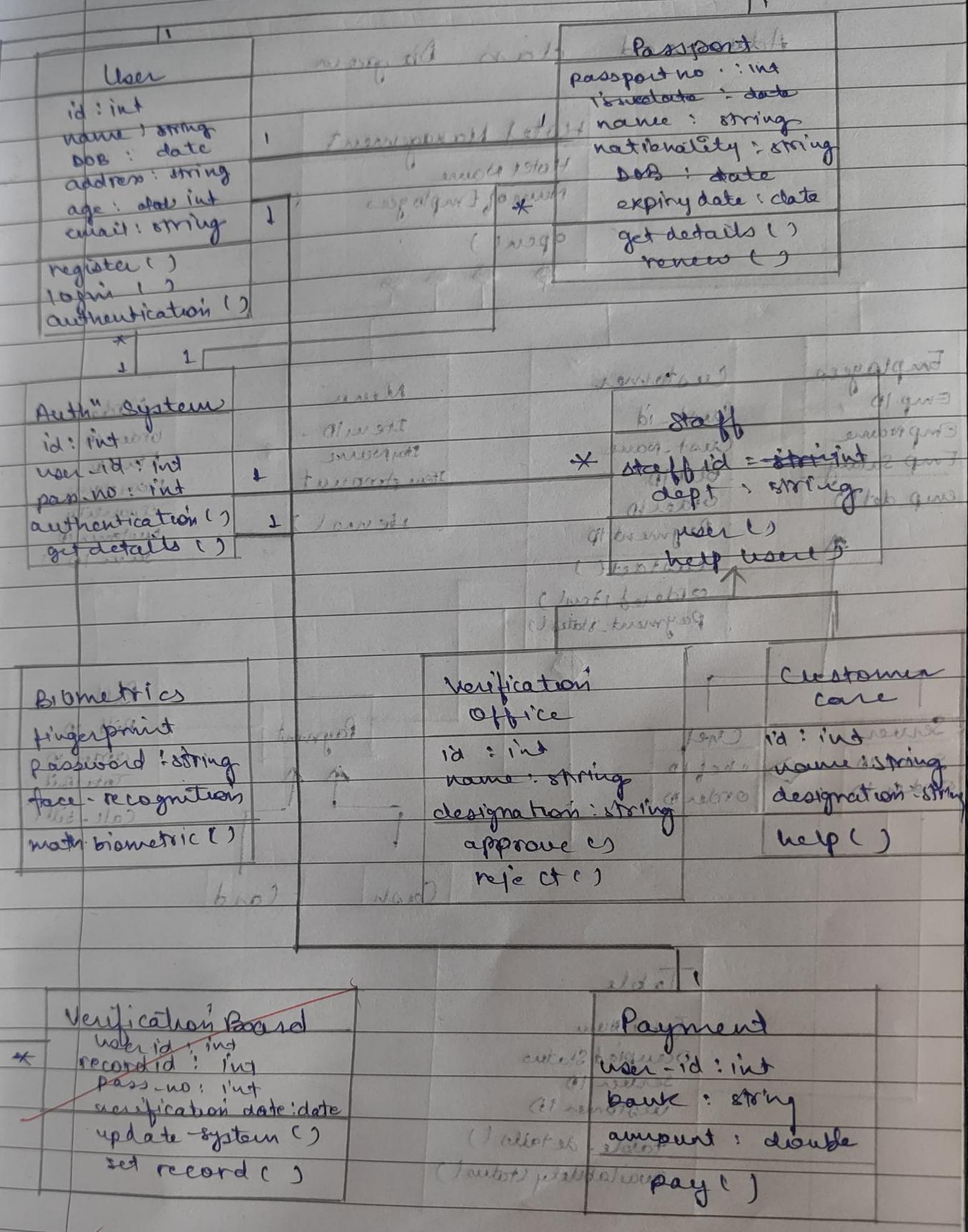
private : exactly
() food won't
() food needed
Order
id : int
date : date
status : string
place order()

Stock Transaction

 id : int
 date : date
 quantity : int
 type : string

```
Customer  
id : int  
name : string  
phone : string  
purchase product()
```

PASSPORT AUTOMATION SYSTEM



X affects

HOTEL MANAGEMENT SYSTEM

Advanced class Diagram

