High level design

Target group: 4th year Computer engineering students

ECE, GIT,DTU

By Misganaw Aguate

MSc. In Computer Engineering

ethiomisgie@gmail.com

YouTube: ethioptech

https://project.ethioptec.com

https://academics.ethioptec.com

High level design

Objectives

The purpose of high-level design

- How a good design lets you get more work done in less time
- Specific things you should include in a high-level design
- Common software architectures you can use to structure an application
- How UML lets you specify system objects and interactions

High level design

Software application Architecture

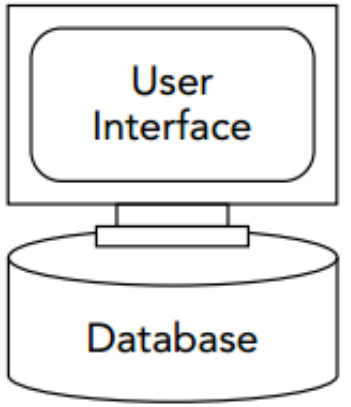
- An application's architecture describes how its pieces fit together at a high level.
- Developers use a lot of "standard" types of architectures.

Monolithic architecture

- In a *monolithic architecture*, a **single program does everything**.
- It displays the user interface, accesses data, processes customer orders, prints invoices, and does whatever else the application needs to do.
- In particular, the pieces of the system are tied closely together, so it doesn't give you a lot of flexibility.
- In this architecture, you understand how all the pieces of the system fit together from the beginning of the project.

High level design

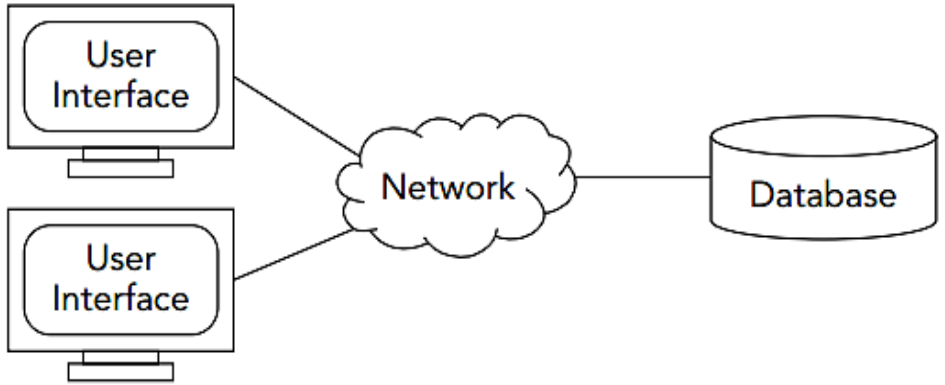
Software application Architecture



Client – server architecture

A *client/server architecture* separates pieces of the system that need to use a particular function (clients) from parts of the system that provide those functions (servers).

- **One problem** with this design is that multiple users **cannot use the same data**.
- You can fix that problem by moving to a *two-tier architecture* where a client (the user interface) is separated from the server (the database).
- The clients and server communicate through some network such as a local area network (LAN), wide area network (WAN), or the Internet.

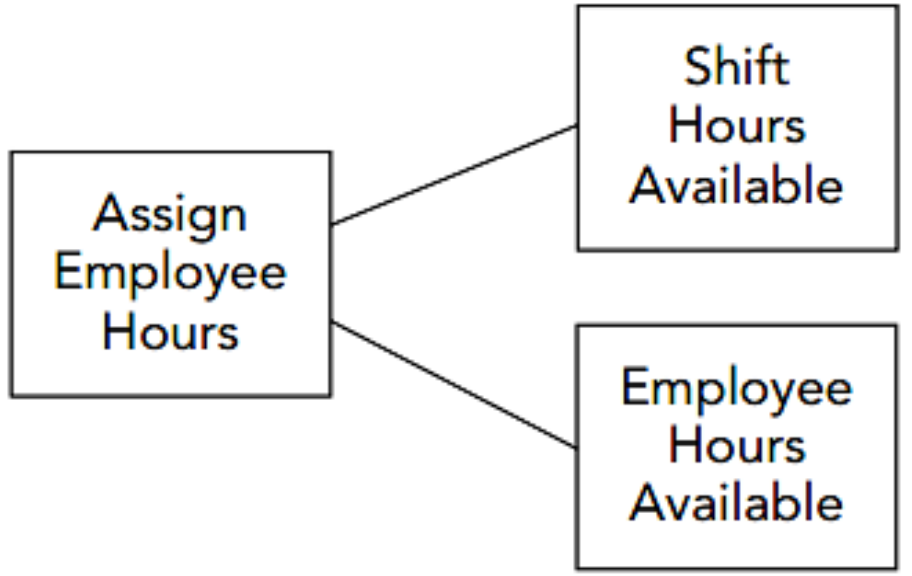


High level design

Software application Architecture

Components based architecture

- *This architecture* regards the system as a **collection of loosely coupled components** that provide services for each other.
- For example, suppose you're **writing a system to schedule employee work shifts**.
- The user interface could dig through the database to see what hours are available and what hours an employee can work
- but that would tie the user interface closely to the database's structure.

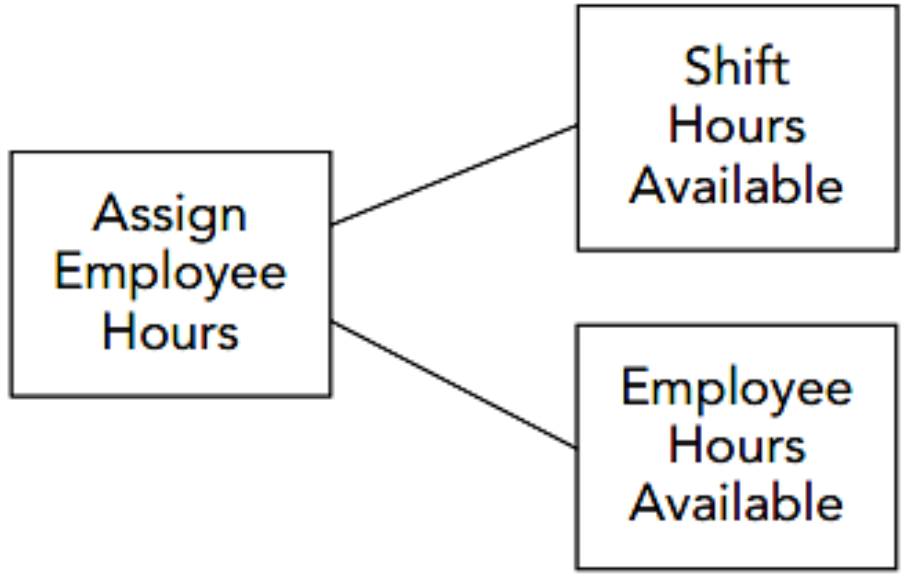


High level design

Software application Architecture

Components based architecture

- The **Assign Employee Hours** user interface component would use the **Shift Hours Available component** to find out what hours were not yet assigned.
- It would use the **Employee Hours Available** component to find out what hours an employee has available.
- After assigning **new hours to the employee**, it would update the other two components so that they know about the new assignment.



High level design

Software application Architecture

Service –oriented architecture

- A *service-oriented architecture* (*SOA*) is similar to a component-based architecture except the **pieces are implemented as services**.
- A *service* is a self-contained program that runs on its own and provides some kind of service for its clients.
- Sometimes, services are implemented as *web services*.
- Those are simply programs that satisfy certain standards, so they are easy to invoke over the Internet.

High level design

Software application Architecture

Distributed architecture

- In a *distributed architecture*, different parts of the **application** run on **different processors** and may run at the **same time**.
- The processors could be **on different computers** scattered across the **network**, or they could be different **cores** on a **single computer**.
- Most modern computers have multiple cores that can execute code at the same time.
- Service-oriented and multitier architectures are often distributed, with different parts of the system running on different computers.

High level design

Software application Architecture

Rule based architecture

- A *rule-based architecture* uses a collection of rules to decide what to do next.
- These systems are sometimes called *expert systems* or *knowledge-based systems*.
- Rule-based systems work well if you can identify the rules necessary to get the job done.
- Sometimes, you can build good rules even for complicated systems; although that can be a lot of work.
- Rule-based systems don’t work well if the problem is poorly defined so you can’t figure out what rules to use.

High level design

- You can view software development as a process that **chops** up the system into **smaller** and **smaller** pieces until the pieces are small enough to implement.
- Using that viewpoint, high-level design is the **first step** in the chopping up process.
- High-level design provides a view of the system at an **abstract level**. It shows how the major pieces of the finished application will fit together and interact with each other.

UML (Unified Modelling Language)

It defines several kinds of diagrams that you can use to represent different pieces of the system.

High level design

Some of UML are

There are many UML diagram. Some of this are

Structure diagram

- Class Diagram
- Composite Structure Diagram
- Component Diagram
- Deployment Diagram
- Object Diagram
- Package Diagram
- Profile Diagram

Behaviour diagram

- Activity Diagram
- Use Case Diagram
- State Machine Diagram
- **Interaction Diagram**

Interaction diagram

- Sequence Diagram
- Communication Diagram
- Interaction Overview Diagram
- Timing Diagram

High level design

Structure diagram

- A *structure diagram* describes things that will be in the system you are designing.
- For example, the class diagram (one type of structure diagram) shows relationships among the classes that will represent objects in the system such as inventory items, vehicles, expense reports, and coffee requisition forms.

Interaction diagram

- *Interaction diagrams* are a subset of activity diagrams.
- They include sequence diagrams, communication diagrams, timing diagrams, and interaction overview diagrams.

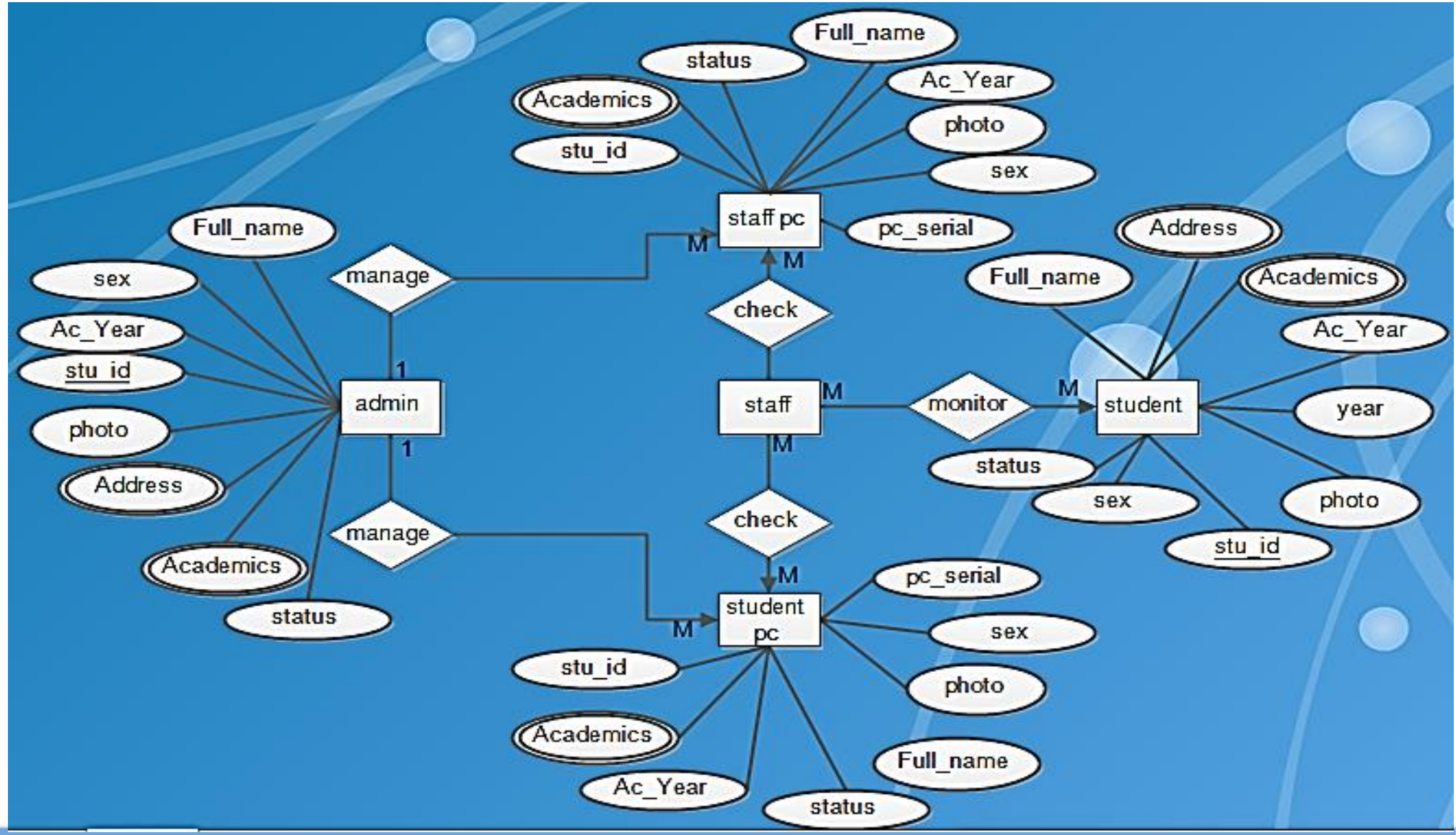
High level design

ER (entity relationship) diagram

- ER *model*, also called an ***entity-relationship*** *(ER) diagram*
- it is a **graphical** representation of **entities** and their relationships to each other
- typically used in computing in regard to the organization of data within databases or information systems.
- An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database.
- An entity in this context is a component of data or concept about which data is stored.

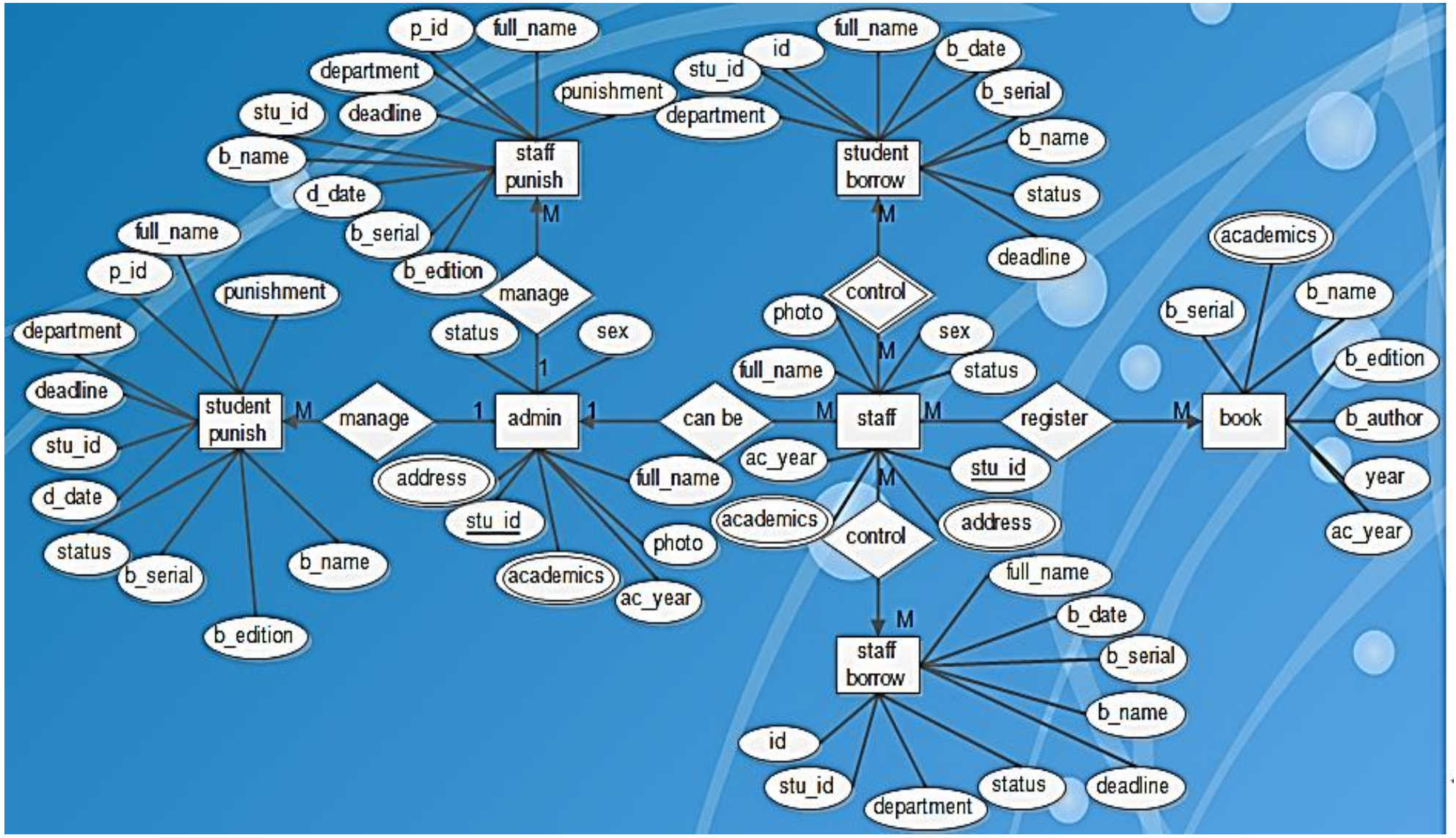
High level design

ER diagram for Outgoing PC Control system



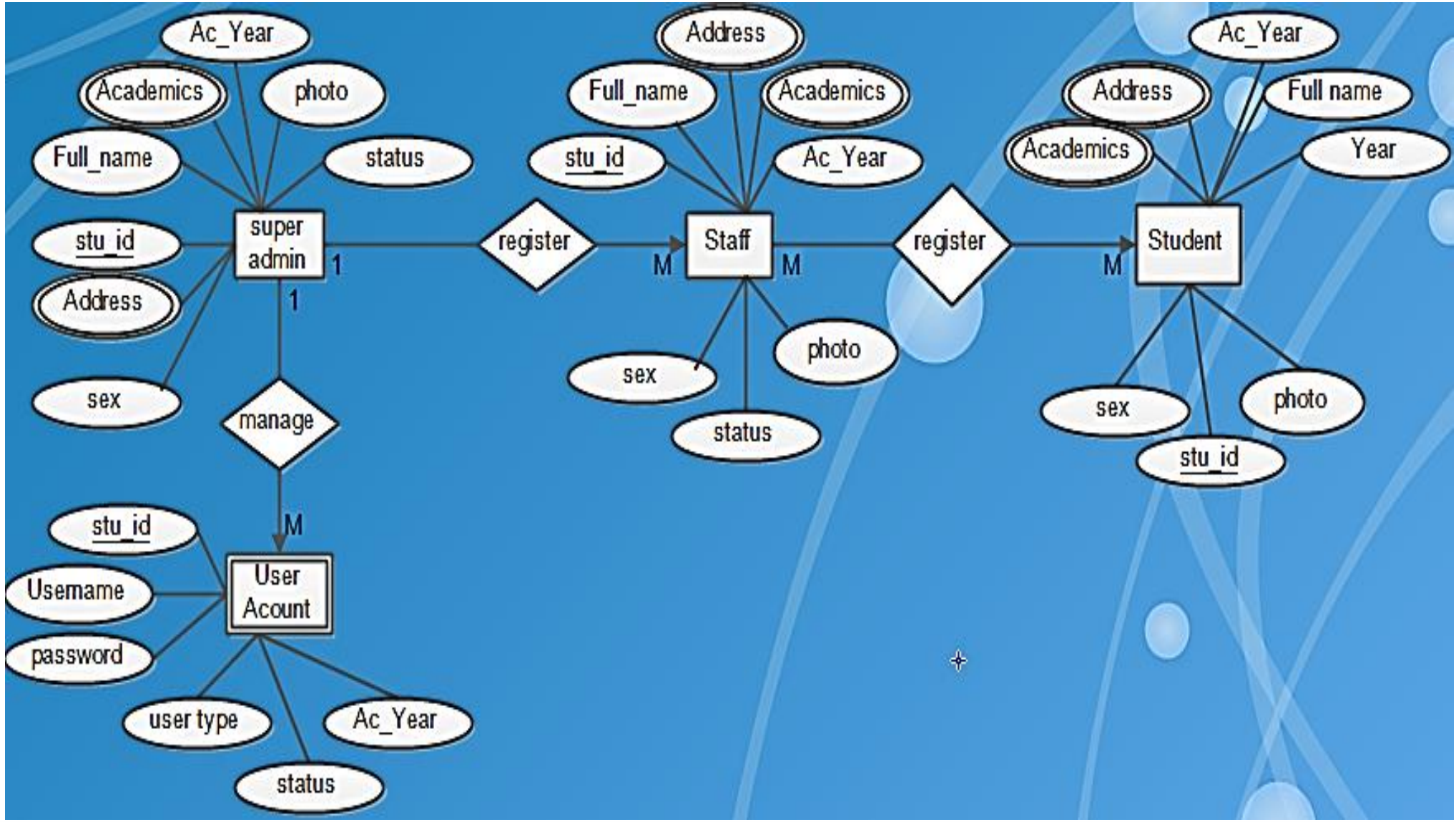
High level design

ER diagram for Library book Borrowing system



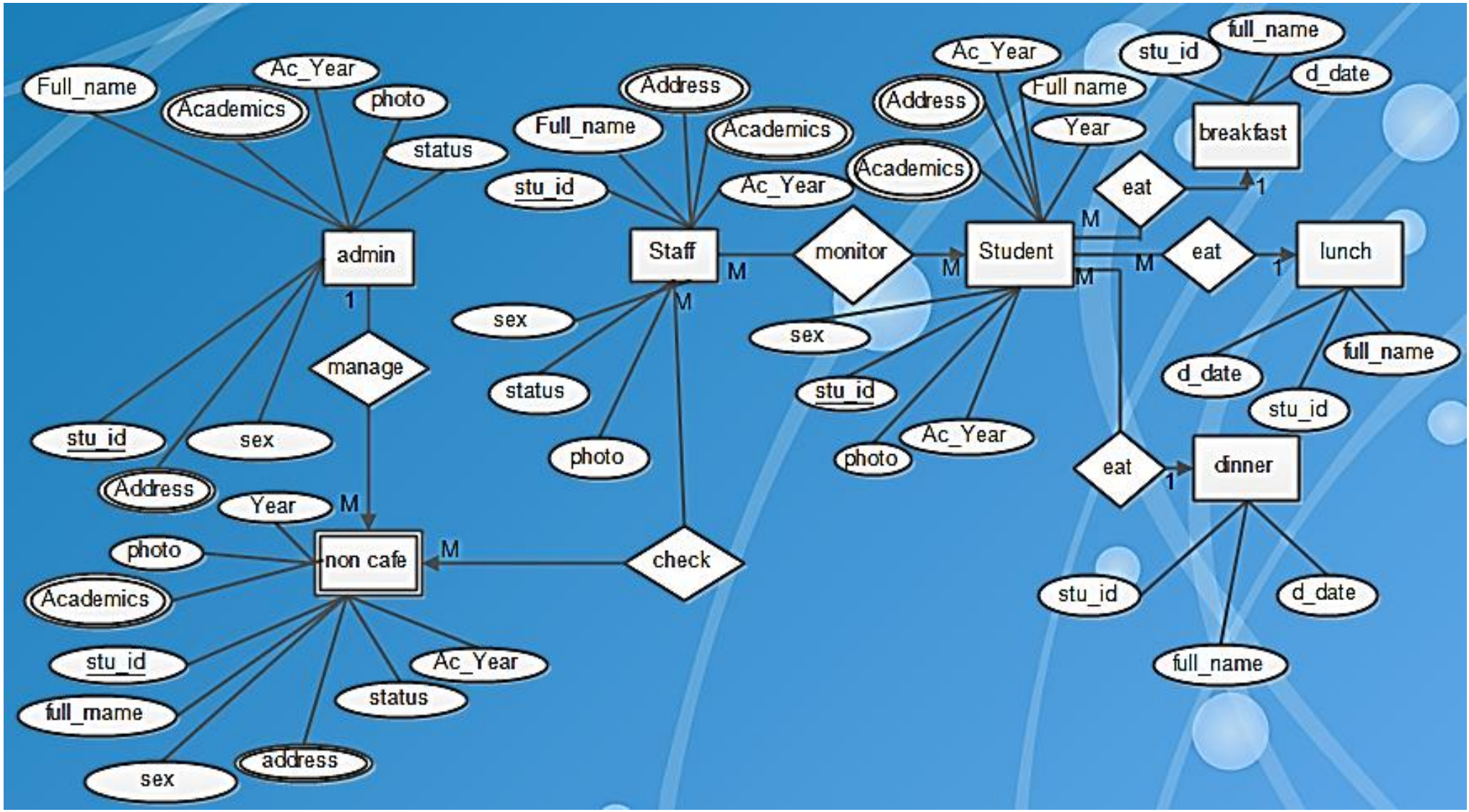
High level design

ER diagram for User registration



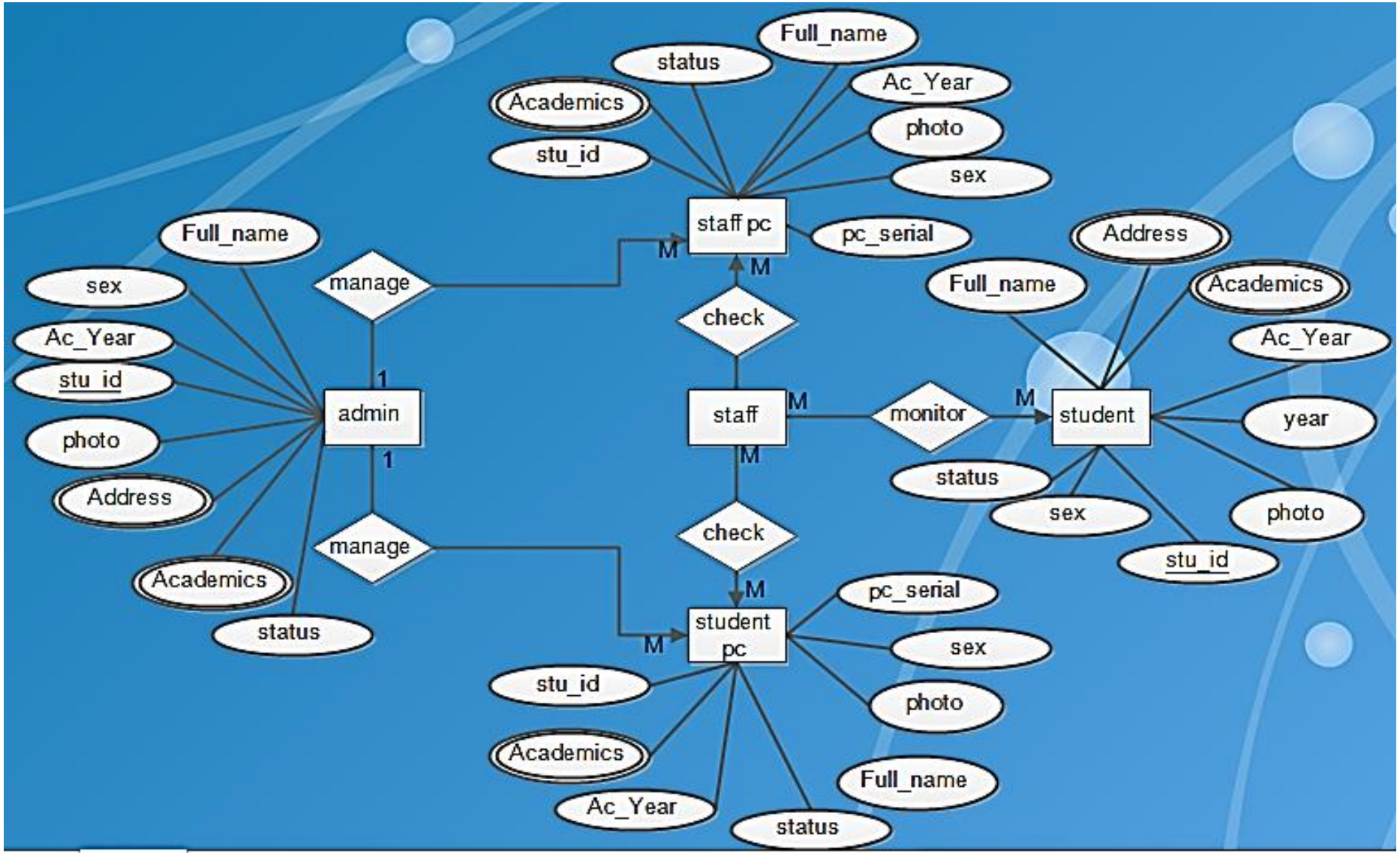
High level design

ER diagram for Meal attendance system



High level design

ER diagram for Outgoing PC Control at the Gateway of Private or Governmental Organization.



High level design

Use case diagram

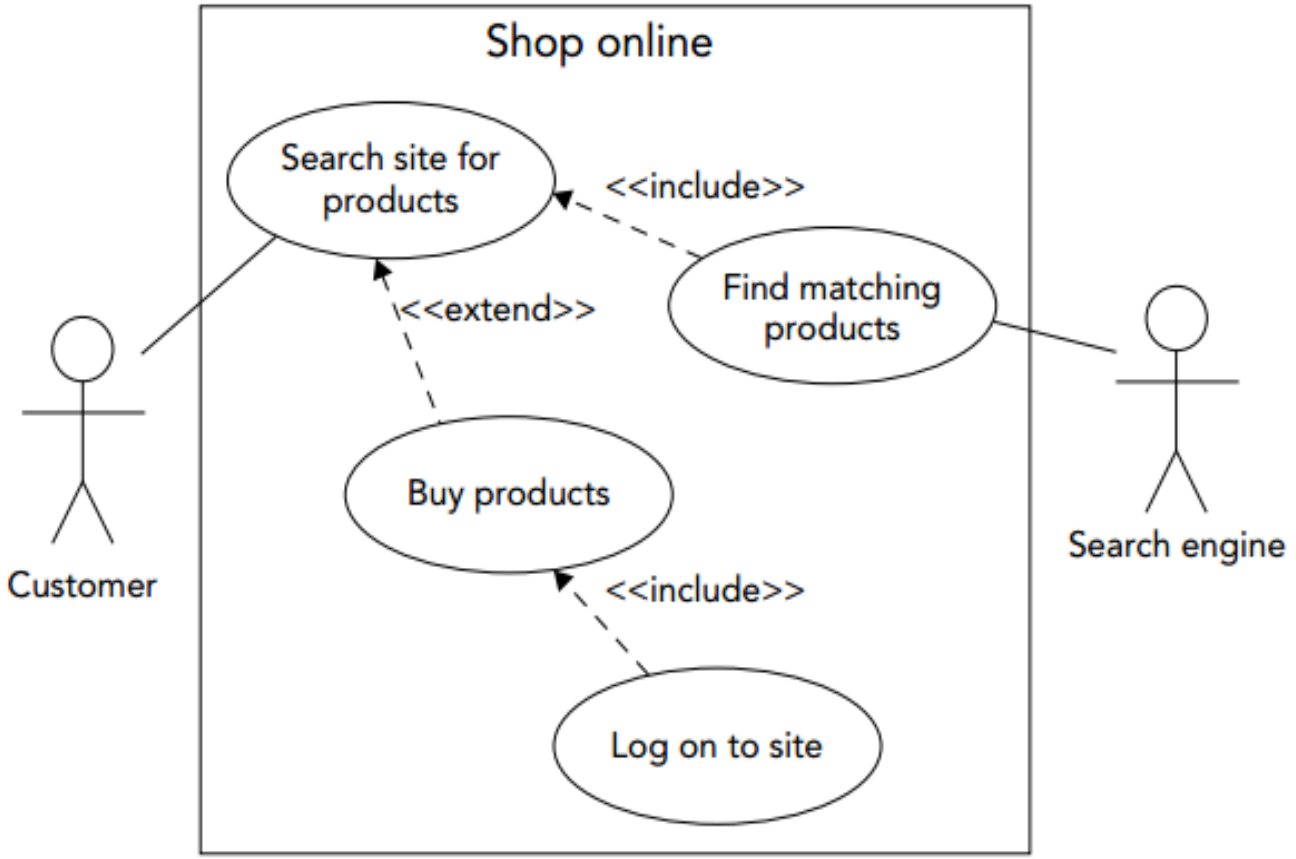
A ***use case diagram*** represents a user's **interaction with the system**.

- Use case diagrams show **stick figures representing actors** (someone or something that performs a task) connected to **tasks represented** by ellipses.
- To provide more detail, you can use **arrows to join subtasks** to tasks.
- Use the annotation **<<include>>** to mean the task includes the subtask.

High level design

Use case diagram

- If a subtask might occur only under some circumstances, connect it to the main task and add the annotation **<<extend>>.**
- If you like, you can add a note indicating when the extension occurs. (Usually both **<<include>>** and **<<extend>>** arrows are dashed.)

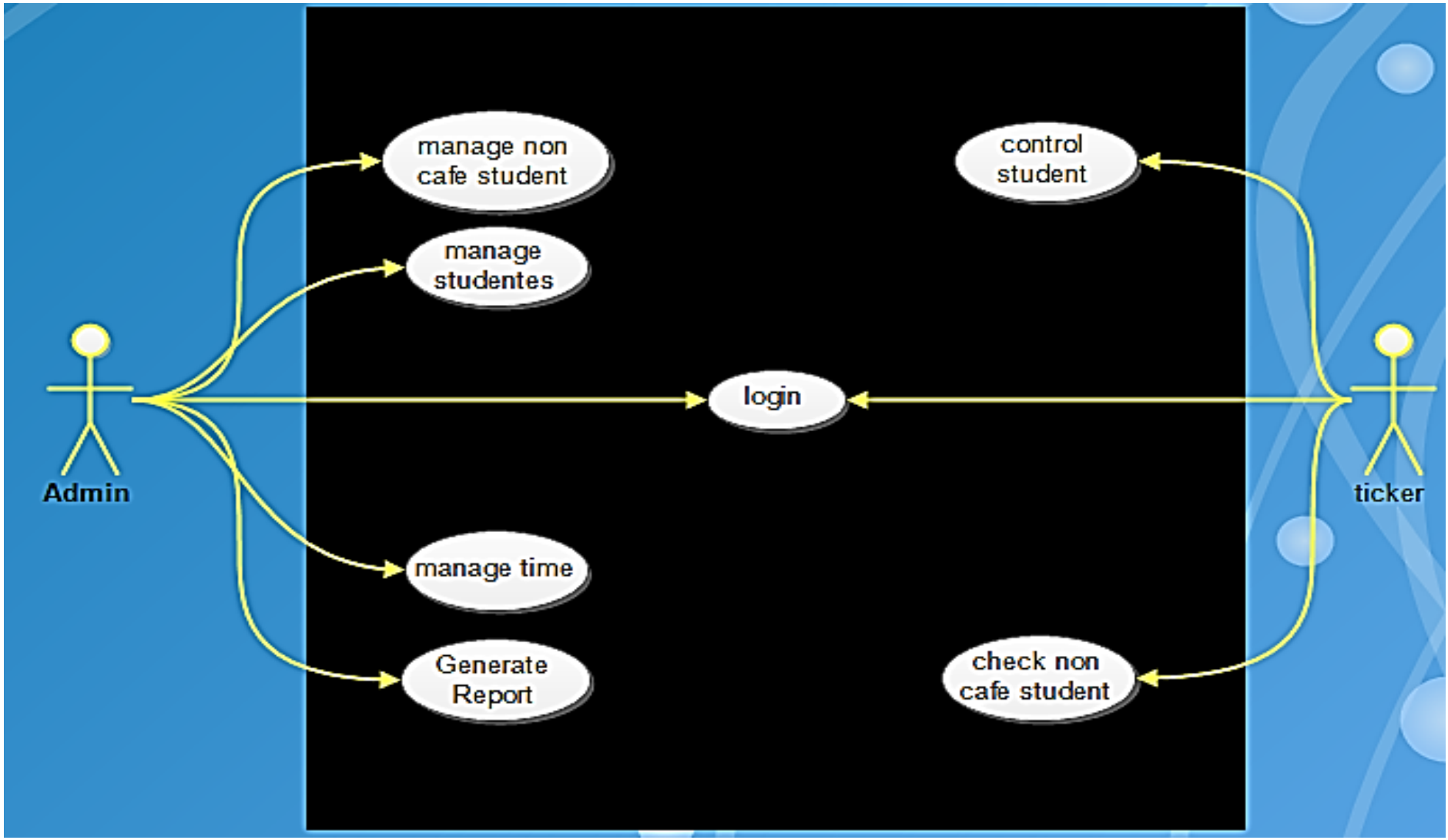


High level design

Use case diagram

Sample use case diagram

For meal attendance system

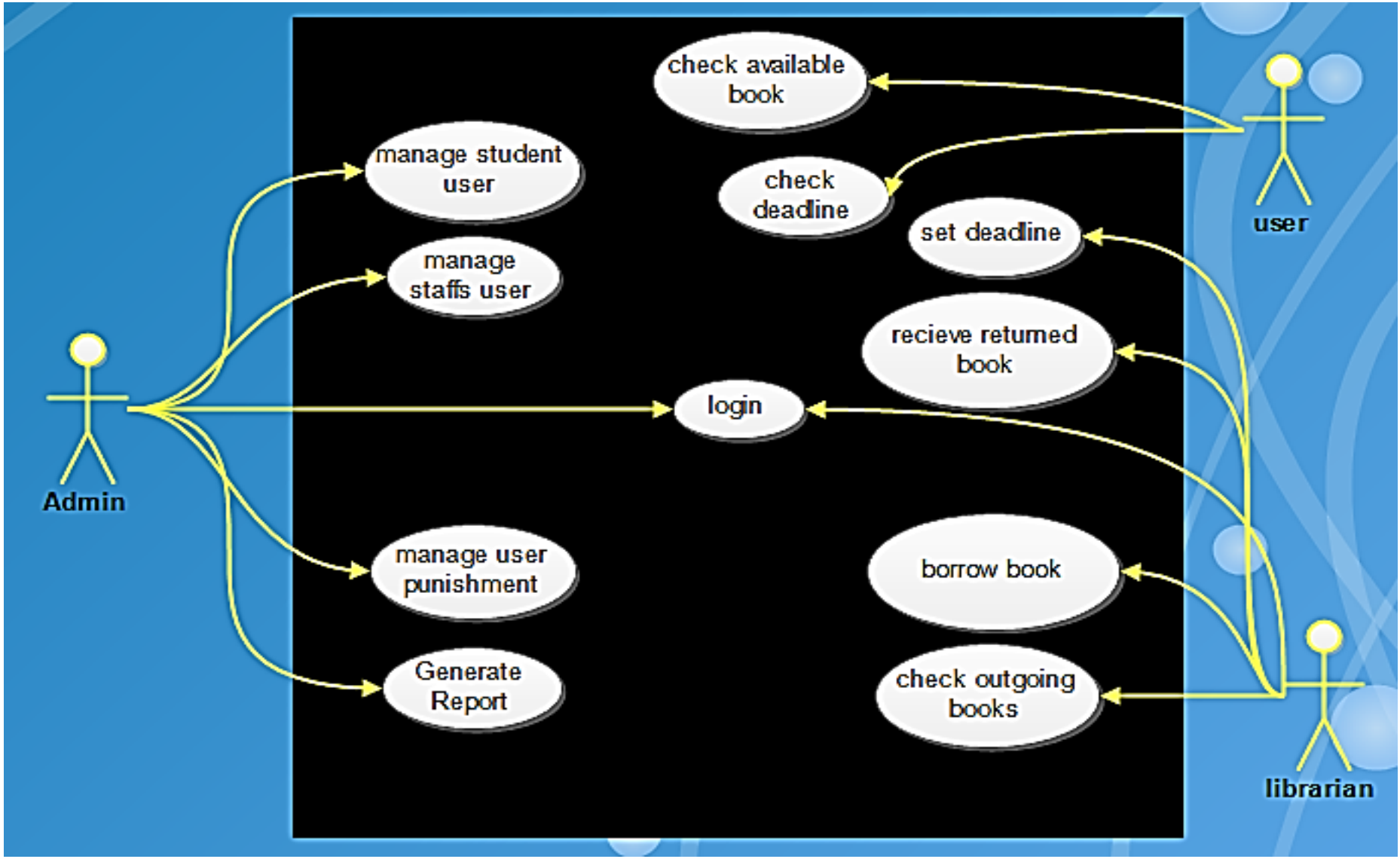


High level design

Use case diagram

Sample use case diagram

For book borrowing system



High level design

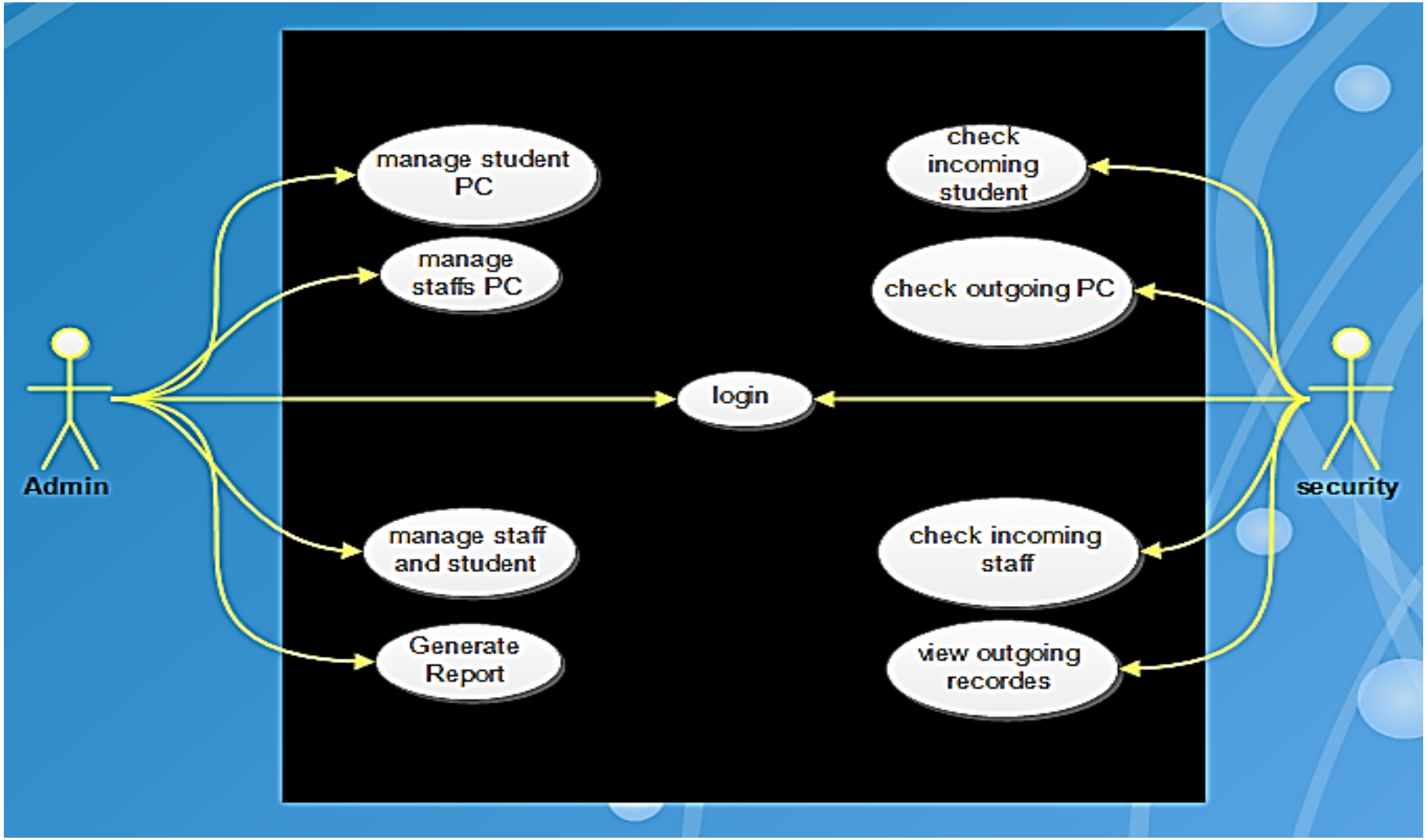
Use case diagram

Sample use case diagram

For outgoing

PC control system

At the gatway



High level design

Sequence diagram

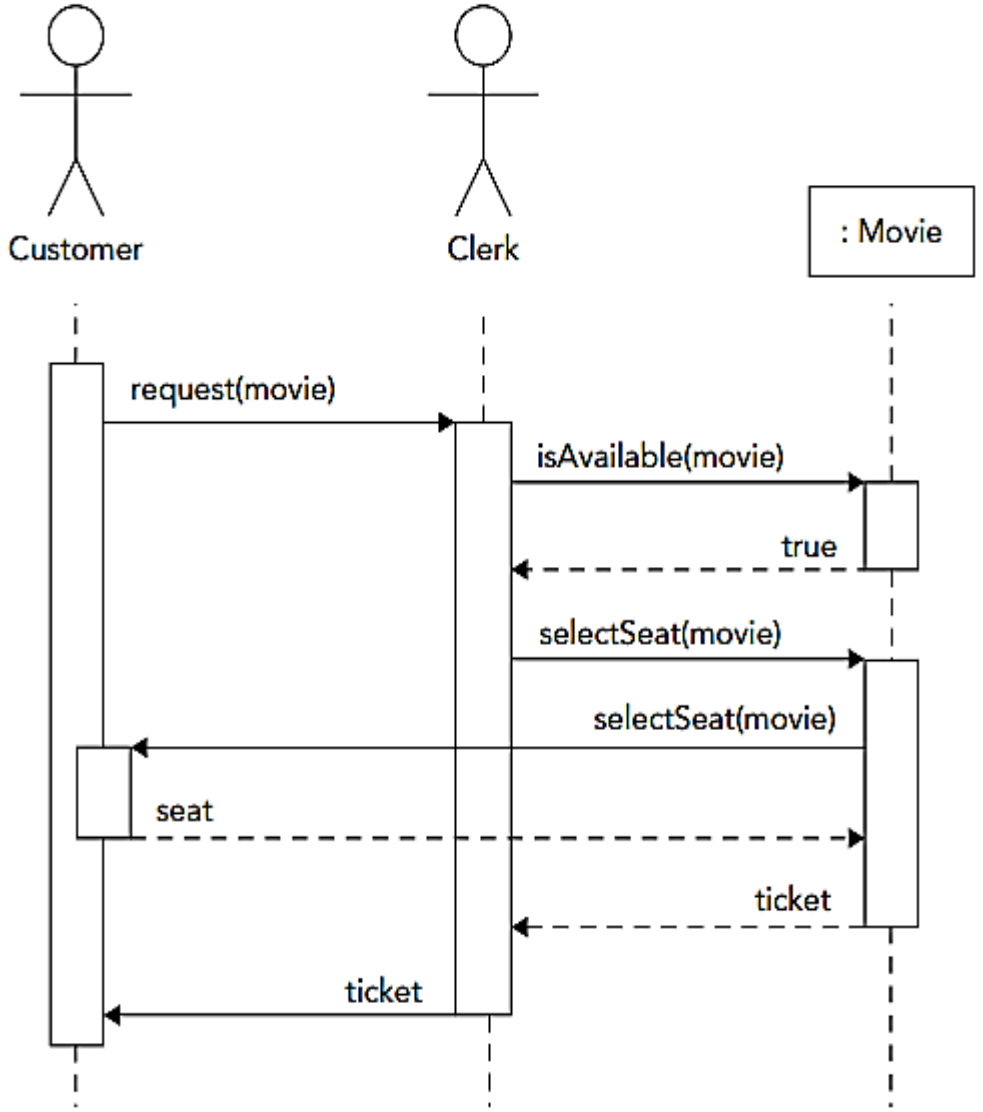
- A *sequence diagram* shows how **objects collaborate** in a particular scenario.
- It represents the **collaboration** as a sequence of messages.
- **Objects** participating in the collaboration are represented as **rectangles** or sometimes as **stick figures** for **actors**. They are labelled with a name or class.
- If the label includes both a **name** and **class**, they are separated by a **colon**.
- Below each of the participants is a **vertical dashed line** called a ***lifeline***. The **lifeline** basically represents the **participant** sitting there waiting for something to happen

High level design

- An ***execution specification*** (called an *execution* or informally an *activation*) represents a **participant doing something**.
- In the diagram, these are represented as **gray** or **white** rectangles drawn on top of the **lifeline**.
- You can draw **overlapping rectangles** to represent overlapping executions.
- Labelled arrows with **solid arrowheads** represent **synchronous** messages.
- Arrows with **open arrowheads** represent **asynchronous** messages.
- Finally, **dashed arrows** with open arrowheads represent **return messages** sent in reply to a calling message.

High level design

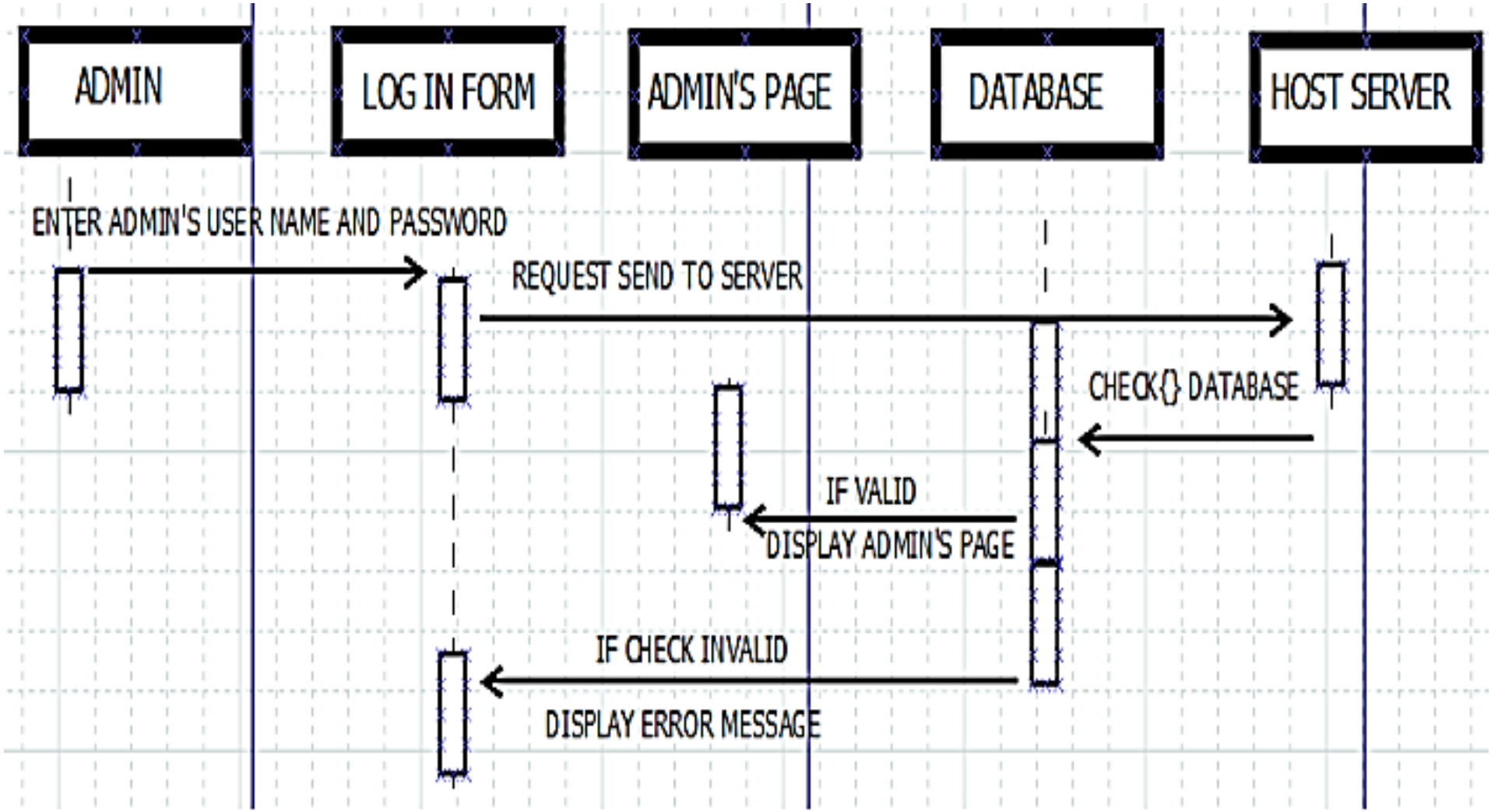
- A customer, a clerk, and the Movie class interacting to print a ticket for a movie.
- The customer walks up to the ticket window and requests the movie from the clerk.
- The clerk uses a computer to ask the Movie class whether tickets are available for the desired show.
- The Movie class responds.



High level design

Sample Sequence diagram

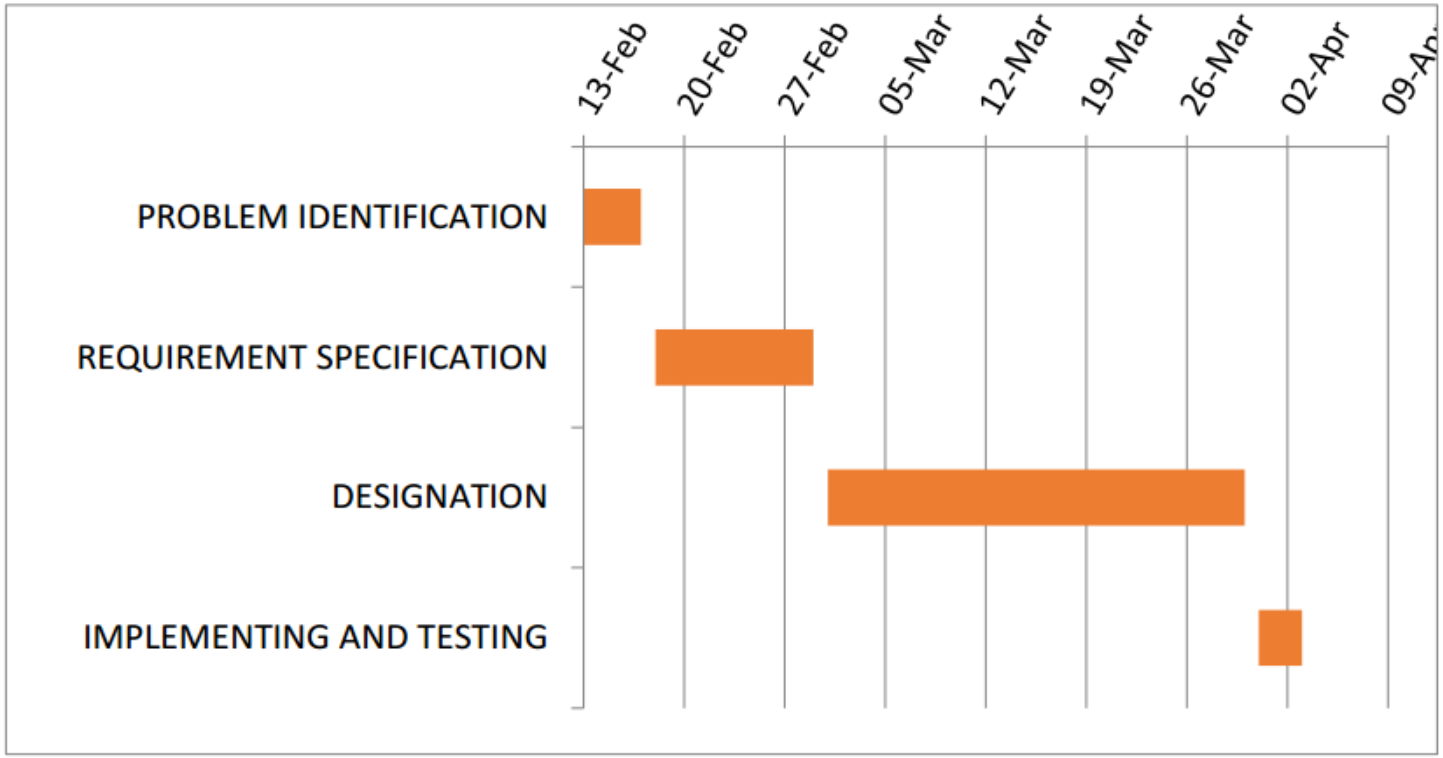
for Login system



High level design

Timing Diagram

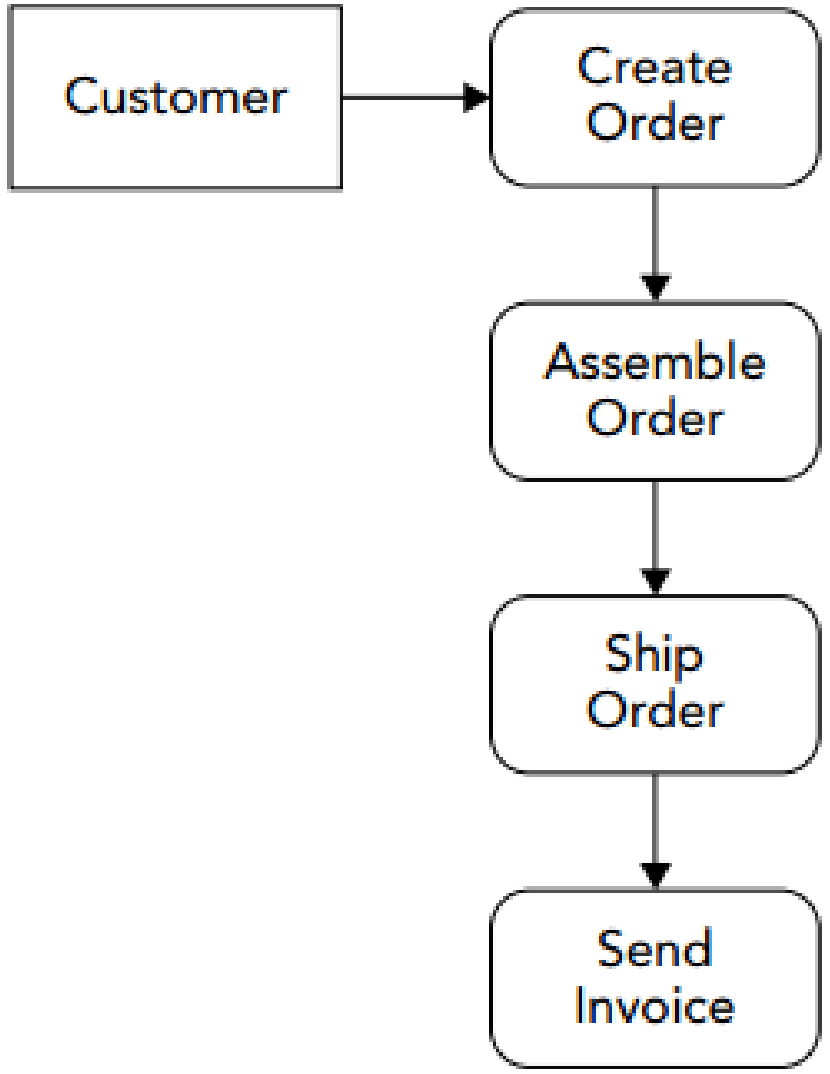
- A *timing diagram* shows one or more objects' changes in state over time.
- A timing diagram looks a lot like a sequence diagram turned sideways, so time increases from left to right.
- These diagrams can be useful for giving a sense of how long different parts of a scenario will take.



High level design

Dataflow diagram

- Many applications use data that flows among different processes.
- For example, a **customer order** might start in an Order Creation process, **move** to Order Assembly (where items are gathered for shipping), and then go to Shipping (for actual shipment).
- Data may flow from Shipping to a final Billing process that sends an invoice to the customer via e-mail.

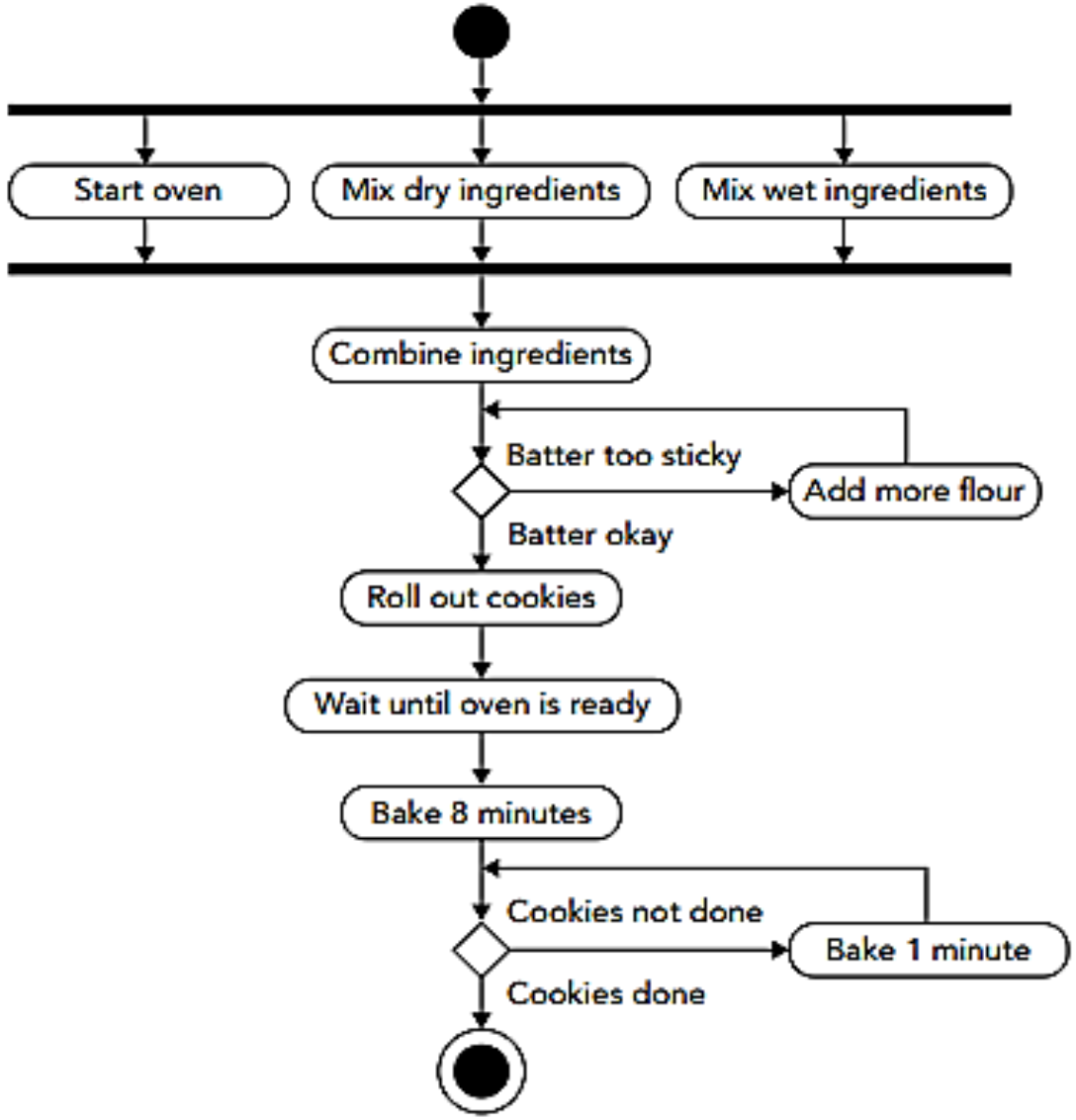


High level design

Activity diagram

- An *activity diagram* represents work flows for activities. They include several kinds of symbols connected with arrows to show the direction of the work flow.

SYMBOL	REPRESENTS
Rounded rectangle	An action or task
Diamond	A decision
Thick bar	The start or end of concurrent activities
Black circle	The start
Circled black circle	The end



Review exercises

Review exercise

1. What's the difference between a component-based architecture and a service-oriented architecture?
2. Suppose you're building a phone application that lets you play tic-tac-toe against a simple computer opponent. It will display high scores stored on the phone, not in an external database. Which architectures would be most appropriate and why?
3. Repeat question 2 for a chess program running on a desktop, laptop, or tablet computer.
4. Repeat question 3 assuming the chess program lets two users play against each other over an Internet connection.

Review exercises

5. Suppose you want to design student attendance system.

- The system will have student registration page to register login detail in tbl_users and student and instructors detail information in tbl_student and tbl_instructors respectively.
- The system will also have attendance page to take attendance that will be stored in tbl_attendance.
- User ID is primary key in tbl_users and foreign key in tbl_student. User ID also primary key in tbl_users and foreign key in tbl_instructor.
- The **id** in tbl_student and tbl_instructor is primary key in this table and foreign key in tbl_attendance and tbl_instructor.
- Student can mark their attendance using take attendance page.
- Instructor can view the absent, present and late student using view report page.
- Instructor can also mark students' attendance, update and delete students from attendance list.
- The system enables Both student and instructor to login into the system. Both can change their login credentials.

Review exercises

- The system will allow students and instructors to access student and instructor page respectively if their username and password provided on login page are correct. If the login password and username are incorrect, the system will redirect the instructor or students to login page.

Draw the following high-level diagram for this system.

- Block diagram of the system (the project).
- ER diagram
- Use case diagram and
- sequence diagram.

Review exercises

Hints for question 5

- **tbl_users** will have user id, student and instructor first name, username and password
- **tbl_student** will have **ID** as primary key, user id as foreign key, student, first, second and last name
- **tbl_instructor** will have **ID** as primary key, user id as foreign key, instructor, first, second and last name
- **tbl_attendance** will have **ID** as foreign key, student first name, second name and last name, absent, present and late, date of attendance columns.

Remember

- **ID** is primary key in tbl_**student** and foreign key in **tbl_attendance.**

Review exercises

6. Suppose you want to design illegal transport service reporting system.

The system will have the following GUI

- passengers sign up page
- Car, owner and driver detail registration
- Traffic police registration
- Login page

The system enables the admin login to the system and manipulate the following activities

- Car, owner and driver registration, modification and deletion
- Traffic police registration, registration, modification and deletion
- Monitoring illegal service reports received from passengers or traffic polices

Review exercises

The system enables the passengers to do the following activities

- Sign up to the system
- Login to the system
- Access car and owner detail
- Report if there occur incident (illegal service)

The system will enable the active traffic police to do the following tasks.

- Login to the system
- View the reported incident by the passengers
- Track the care that the reported incidents are occurred

Review exercises

The system will have the following tables in the database

- Tbl_users
- Tbl_passangers
- Tbl_cars
- Tbl_owners
- Tbl_drivers
- Tbl_trafficpolice
- Tbl_incident
- Tbl_admin

Review exercises

Tbl_users table will have columns of user_id, username and password.

The user_id is the foreign key that relates

- the tbl_passangers using passanger_id. passanger_id in tbl_passangers is the primary key. But in tbl_users, it is a foreign key found as user_id.
- the tbl_drivers using driver_id. driver_id in tbl_drivers is the primary key. But in tbl_users, it is a foreign key found as user_id.
- the tbl_owners using owner_id. owner_id in tbl_owners is the primary key. But in tbl_users, it is a foreign key found as user_id.
- the tbl_trafficpolice using trafficpolice_id. trafficpolice_id in tbl_ trafficpolice is the primary key. But in tbl_users, it is a foreign key found as user_id.

Review exercises

- Tbl_passangers will have columns of passangers_id as primary key, full_name, mobile_number.
- Tbl_care will have columns of car_id, owner_id, driver_id and type. Owner_id is foreign key to relate tbl_car with tbl_owner. Driver_id is also foreign key to relate tbl_car with tbl_driver.
- Tbl_ownres will have columns of owner_id as primary key, full_name, mobile_number.
- Tbl_driver will have columns of driver_id as primary key, full_name, mobile_number
- Tbl_trafficpolice will have columns of trafficpolice_id, full_name, mobile_number
- Tbl_admin will have colomuns of admin_id, full_name, mobile_number
- Tbl_incident will have columns of incident_id as primary key, car_id as foreign key, incident_type, incident_description, incident_date, departure_city, arrival_city.

Review exercises

Draw the following high-level diagram for this system.

- Block diagram of the system (the project).
- ER diagram
- Use case diagram and
- sequence diagram.

The system will allow passengers, admin, traffic and driver to access their respective page if their username and password provided on login page are correct. If the login password and username are incorrect, the system will redirect the passengers, drivers, owners, traffic polices or admins to login page.

Review exercises

7. Suppose you want to design Library book borrowing system.

The system will have the following pages

Library admin page

Book registration and borrowing page

Student page

Lecturer page

Report viewing and preparation page

Review exercises

Library admin can do the following activities

- Login to the system
- Register, update, and delete student data
- Register, update, and delete lecturer data
- Register, update, and delete book data
- View and prepare reports
- Make student and lecturers lend book

Student and lecturer can do the following tasks

- View available books in the library, view deadline (the date that the book to be returned)
- Reserve books and lend

Review exercises

The system will have the following tables

Tbl_users

Tbl_students

Tbl_lecturer

Tbl_admin

Tbl_book

Tbl_borrow

Tbl_return

Tbl_users will have id as primary key, **user_id**, username and password. **User_id** is **the foreign key** that relates tbl_users with

- **Tbl_students** that has colomns of student_id as primary key, full_name, department and phon_number. Student_id is user_id in tbl_users.
- **Tbl_lecturer** that has colomns of leturer_id as primary key, full_name, department and phone_number. lecturer_id is user_id in tbl_users.
- **Tbl_admin** that has colomns of admin_id as primary key, full_name and phone_number. admin_id is user_id in tbl_users.

Review exercises

- Tbl_book will have book_id, name, autor_name, edition, department, target_year. Book_id is primary key.
- Tbl_borrow will have id as primary key, borrowing_id as foreign key, book_id as foreign key, borrowing_date, deadline. Borrowing_id used to relate tbl_borrow with tbl_student or tbl_lecturer. That means borrowing_id may be student_id or lecturer_id. Book_id used to relate tbl_borrow with tbl_book.
- Tbl_return will have id as a primary key, borrowing_id as foreign key, book_id as foreign key. Borrowing_id used to relate tbl_return with tbl_student or tbl_lecturer. That means borrowing_id may be student_id or lecturer_id. Book_id used to relate tbl_borrow with tbl_book.

Review exercises

Draw the following high-level diagram for this system.

- Block diagram of the system (the project).
- ER diagram
- Use case diagram and
- sequence diagram.

The system will allow students, lecturers and admin to access their respective page if their username and password provided on login page are correct. If the login password and username are incorrect, the system will redirect the instructor, students or admin to login page.

Review exercises

8. Let you are going to develop e-commerce

The system will have the following pages

- Product page to display the product.
- Cart page to display the selected product to be purchased.
- Payment page that makes the purchaser pay for the selected products available in the cart.
- Confirmation page to check whether the payments of ordered products are completed or not.
- Login page to enable customer and seller login to the system

Review exercises

The system will have the following functionalities

- It enable seller to post products on the system, update posts and delete posts
- The system enable the customer to view products, add to cart, pay for the added product in cart and track the ordered products.

Thes system will have the following tables in the database

- Tbl_users that has columns of id as primary key, user_id as foreign key, username and password. User_id used to relate tbl_seller and tbl_customer. Seller_id from tbl_sell and customer_id from tbl_customer will be taken as user_id in tbl_user
- Tbl_seller that has columns of seller_id as primary key, full_name, email and phone_number, bank_account.

Review exercises

- Tbl_customer will have columns of customer_id as primary key, full_name, phone_number, bank_account.
- Tbl_product will have columns of product_id as primary key, name, brand_name, product_category, price.
- Tbl_cart will have columns of cart_id as primary key, product_id as foreign key, customer_id as foreign key, quanity, total_price. Product_id and customer_id are used to relate tbl_cart with tbl_product and tbl_customer respectively.
- Tbl_payment will have columns of payment_id as primary key, cart_id as foreign key, paymet_status.
- Cart_id will used to relate tbl_payment with tbl_cart.
- This helps to know who purchased the product, what is the quaintly of purchased product and cross checke whether the payment is completed or not.

Review exercises

Draw the following high-level diagram for this system.

- Block diagram of the system (the project).
- ER diagram
- Use case diagram and
- sequence diagram.

The system will allow seller and customer to access their respective page if their username and password provided on login page are correct. If the login password and username are incorrect, the system will redirect the seller or customer to login page.

Review exercises

9. Let you want to design patient appointment and follow up system that will have the following pages and functionalities

Pages

- Login page
- Patients page
- Admin page:
- Doctors page
- Recorder page and
- Pharmacist page

Review exercises

Users of the system

Patients

- Patients can do the following activities
- View their treatment history including type of disease they encountered and respective remedial drugs
- View appointment date
- Report symptoms if necessary

Doctors

- Doctors can view the reported symptoms from the patients and give appointment for next follow up.
- Register patient history and order appropriate laboratory test and medicine (drug)

Review exercises

Recorder

Recorder can register patient information including record number

Pharmacist

- Pharmacist can view the patient history settled by doctor
- According to the history, the pharmacist will give drug for the patient

Review exercises

the system will have the following tables

Tbl_users: has id, user_id, username and password columns

Tbl_patients: has p_id, full name, address, age and gender columns

Tbl_doctors: has d_id, full name and specialization columns

Tbl_pharmacist: has pha_id, full name and specialization columns

Tbl_recorder: has r_id, full name

Tbl_p_history: has id, patient_id, doctor_id, disease, ordered drug and utilization instruction columns

Tbl_appointment : has id, patient_id, doctor_id and date

Review exercises

- user_id in tbl_users area foreign key that relate this table with tbl_ptients, tbl_doctors,tbl_pharmacist and tbl_recordor. User_id in tbl_users are p_id, d_id,pha_id,r_id that are primary key of tbl_patients, tbl_doctors, tbl_pharmacist and tbl_recordors respectively.
- Patient_id and doctor_id in tbl_history are foreign keys that relate this table with tbl_patient and tbl_doctor respectively. Patient_id and doctor_id are p_id and d_id that are primary keys in tbl_patients and tbl_doctors. This relation are required to know which patient history is recorded by whom doctor and which patient history is belongs to whom patients.
- Patient_id and doctor_id in tbl_apoinment are foreign keys that relate this table with tbl_patient and tbl_doctor respectively. Patient_id and doctor_id are p_id and d_id that are primary keys in tbl_patients and tbl_doctors. This relation is required to know what patients are appointed by whom doctor.

Review exercises

Draw the following high-level diagram for this system.

- Block diagram of the system (the project).
- ER diagram
- Use case diagram and
- sequence diagram.

The system will allow admin, doctors, recorders, pharmacist and patients to access their respective page if their username and password provided on login page are correct. If the login password and username are incorrect, the system will redirect to their respective login page.

End of Chapter 4