CSE1005 Software Engineering Laboratory

<u>Crime Record Management System</u>

SUBMITTED BY :

P. Dhanush Siva Sai Chandranath

20BCI7319

VITAP UNIVERSITY

AMARAVATI

INTRODUCTION

Crime Records Management System is implemented at all Police Stations across the country and specifically focuses on the prevention and detection of criminal activity, as well as the conviction of criminals, which depends on an extremely responsive Information Management system.The quality of information the Police can glean from its existing records determines the efficiency and effectiveness of police work.

OBJECTIVE

Our project's main idea is to develop an online application for improving the complaint system online for a police department. This application will help citizens to file a complaint through a website which will be a time saving and fast problem solving method.Crime Record Management System applications will be implemented in every police station all over the country which will concentrate on handling complaints, prevention of crime by interconnecting police information systems to different police stations in the country. Using this application information handling will be easy and fast and solving cases will be fast.

PROBLEM STATEMENT

The workflow in CRIME RECORDS MANAGEMENT SYSTEM is to maintain all the required applications from customers, details of employees and customers in the form of papers or files. A big problem arises, if those papers are lost, as they are essential for all the work to get done. Reconstruction of all the lost work requires much time and more man power. In order to make a perfect and secured workflow, we have to use a new way, which is completely online. This should include all the required tasks like Customers applying for registration, adding complaints and employees issuing all the status to customers who have launched a complaint. Admin has the details of all the citizens he gives the tasks to do.
To design this site, Security issues are to be considered with care, providing complete security to alL the users and data is essential

PROPOSED SYSTEM

In the proposed system, the aim of the project is to bring about improvement to the organization’s contribution.

Advantages of the Proposed System:

1. Every individual has an unique id and he/she can access the data and status of the crime by sign-in account.
2. We can add notifications to the admin such as update or remove the stations which are notresponding.
3. The product provides a framework within which a user can easily communicate with thestation manager and administration.
4. Complete separation of the access data based on roles: Citizens Stations and Administrator.
5. Ensures data access authentication.
6. Dynamically updating the crime and status of the complaints.
7. We can include feedback from the citizens.
8. We can include a search option for the complete crime details.
9. Avocation is implemented in the proposed system.

EXISTING SYSTEM

We have an existing system where only users are allowed to see certain details about our police stations; however, it poses a lot more work to the authorized person. There are a lot of records and files to be dealt with in this system.As previously developed, the Crime Records Management system only deals that when a citizen enters an FIR number then the system only displays whether the case is cleared or not but it does not provide the complete details of the case.

In the existing system,

1. The existing system is time consuming.
2. It does not provide authentication for the citizens.
3. Many cases are piled up in the corners, which are not proved and which may be a path for the newly added cases which cannot be gathered together.
4. The citizens cannot register their complaints online.
5. In the existing system there are no Avocation facilities.

SOFTWARE DESCRIPTION : The tools that I am going to use is

1. Frontend/language – HTML,CSS,JS
2. Backend/ database - JSP
3. Additional Tools – ORACLE
4. Operating System – Windows 10.

 JSP : Java Server Pages

● SQL : Structured Query Language ●
RBAC : Role Based Access Control

JSP

JSP is a Web application framework developed and marketed to allow programmers to build dynamic Websites, Web applications and Web services.

SQL SERVER

A database is a type of software that stores and retrieves data when requested by other software applications, whether they're running on the same computer or across networks (including the Internet).

ROLE BASED ACCESS CONTROL

Role Based Access Control is a database through which the data is accessed according to the roles of the users who are accessing the database to retrieve the data.

ABBREVIATIONS

HTTP :	Hypertext Transfer Protocol
HTML :	Hyper Text Markup Language
URL :	Uniform Resource Locator
SRS :	Software Requirement Specification
WWW :	World Wide Web
CTO :	Central time officer

IDENTIFYING THE REQUIREMENTS FROM THE PROBLEM STATEMENTS(SRS):

FUNCTIONAL REQUIREMENTS:

It is the primary requirements that are fulfilled by our ONLINE CRIME REPORTING SYSTEM .It's allowing the The registered users, or current The registered user to use our website at the level of ease .The purpose of our website is to provide the full information that is required to the The registered user. Here is the following requirement that is fulfilled by our system.

The registered user

The registered user Login

This feature used by the The registered user/admin to login into system. A The registered user/admin must login with his The registered user name and password to ONLINE CRIME REPORTING SYSTEM after registration. If they are invalid, the The registered user not allowed to enter ONLINE CRIME REPORTING SYSTEM.

- The registered user name and password will be provided after The registered user registration is confirmed.
- Password should be hidden from others while typing it in the field

Register New user

A new The registered user will have to register in ONLINE CRIME REPORTING SYSTEM by providing essential details in order to view the products in ONLINE CRIME REPORTING SYSTEM. The admin must accept a new The registered user by unblocking him. System must be able to verify and validate information.

- ONLINE CRIME REPORTING SYSTEM must encrypt the password of the customer to provide security.

Comlpain History

The registered user user can add the desired complain into his cart by clicking add to cart option on the product. He can view his cart by clicking on the cart button. All products added by cart can be viewed in the cart. The registered user can remove an item from the cart by clicking remove. After confirming the items in the cart the The registered user can submit the cart by providing a delivery address. On successful submitting the cart will become empty.

– System must ensure that, only a registered customer can purchase items.

Search Products

The user can search the desired product. He can view description of product. After confirming the items in the search The registered user can select it into cart by providing a delivery address.

Upload Prescription

The The registered user can upload the prescription if The registered user cannot read the doctor prescription.

Admin

Manage The registered user

The administrator can add The registered user, delete The registered user, view The registered user.

Manage Police Station

The administrator can add product, delete product, hide product and view product. **Manage Updation**

The administrator can view orders and update complaint status.

-ONLINE CRIME REPORTING SYSTEM must identify the login of the admin.

-Admin account should be secured so that only owner of the account can access that account

NON-FUNCTIONAL REQUIREMENTS

Efficiency Requirement

When an online CRIME REPORTING SYSTEM OF POLICE STATION implemented The registered user can create complaint in an efficient manner.

Reliability Requirement

ONLINE CRIME REPORTING SYSTEM should provide a reliable environment to both The registered users and administrator . All complaint should be reaching at the admin end without any errors.

Usability Requirement

ONLINE CRIME REPORTING SYSTEM is designed for The registered user friendly environment and ease of user.

Implementation Requirement

Implementation of ONLINE CRIME REPORTING SYSTEM using CSS, AJAX and html in front end with PHP as back end and it will be used for database connectivity. And the database part is developed by Xampp. Responsive web designing is used for making the website compatible for any type of screen.

Delivery Requirement

The whole system is expected to be delivered in 6 months of time with weekly evaluation by the project guide.

Database Security

Unauthorized person cannot access the panel and database, do not read and write the information.

Availability

ONLINE CRIME REPORTING SYSTEM will be available Monday-Friday(24-Hours),Saturday (24-Hour) and Sunday 4.30am-12.00am.

FUNCTIONAL REQUIREMENTS

OperatingSystem	:Windows10
Technology	:Java/J2EE(Servlets,JSP,JDBC
)WebTechnologies	:Html,JavaScript,CSS
WebServer	:Tomcat7.0
Database	:Oracle10gExpressEdition
Software's	:JDK1.6

NON-FUNCTIONAL REQUIREMENTS

Hardware	:PentiumbasedsystemswithaminimumofP4
RAM	:256MB(minimum)

ESTIMATION OF PROJECT METRICS:

One of the major activities of the project planning phase, therefore, is to estimate various project parameters in order to take proper decisions. Some important project parameters that are estimated include:

Project size: What would be the size of the code written say, in number of lines, files, modules?

Cost: How much would it cost to develop a software? A software may be just pieces of code, but one has to pay to the managers, developers, and other project personnel.

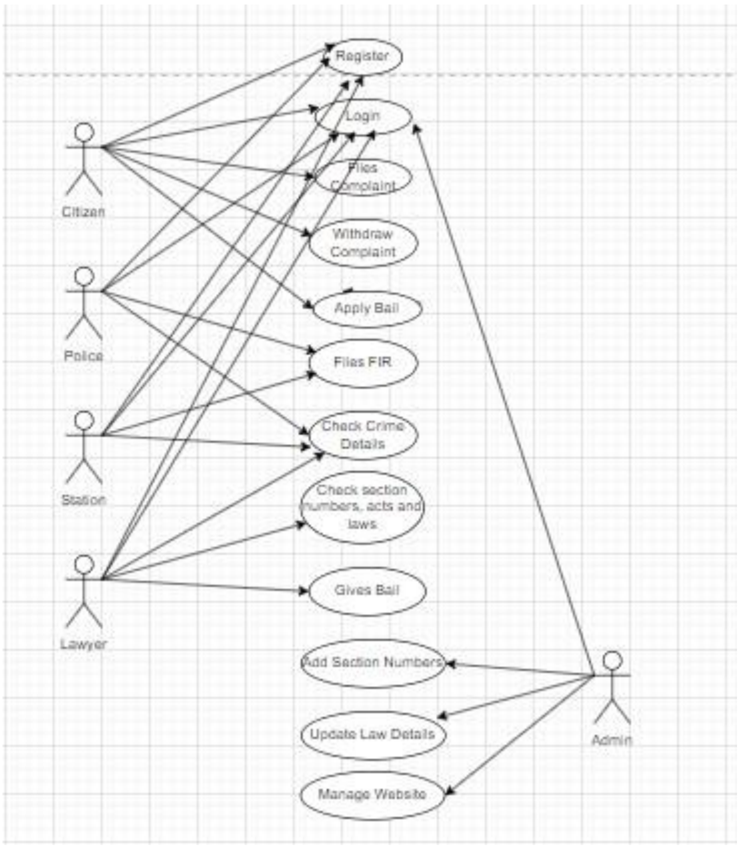
Duration: How long would it be before the software is delivered to the clients?

Effort: How much effort from the team members would be required to create the software?

MODELING UML USER CASE DIAGRAMS:

Use case diagram is valuable for visualizing the practical requirements of the system that will translate into design choices and development priorities. Use case diagrams are valuable UML diagram type and commonly used to analyse various systems. They enable you to imagine the different types of roles in a system and how those roles cooperate with the system.

The below figure shows the use case diagram for Criminal record management system where the admin and sub-admin can use the software for adding criminal, viewing criminal list, adding crime report, viewing crime report and creating accounts, on the other part police can also add criminals and crime reports, viewing criminals lists and crime reports, where the court can use the software for giving punishments to the criminals



MODELING UML CLASS DIAGRAMS:

Class Diagram:

It is a graphical representation for describing a system in context of its static construction.

Elements in class diagram

Class diagram contains the system classes with its data members, operations and relationships between classes.

Class

A set of objects containing similar data members and member functions is described by a class. In UML syntax, class is identified by solid outline rectangle with three compartments which contain

- Class name

A class is uniquely identified in a system by its name. A textual string [2]is taken as class name. It lies in the first compartment in class rectangle.

- Attributes

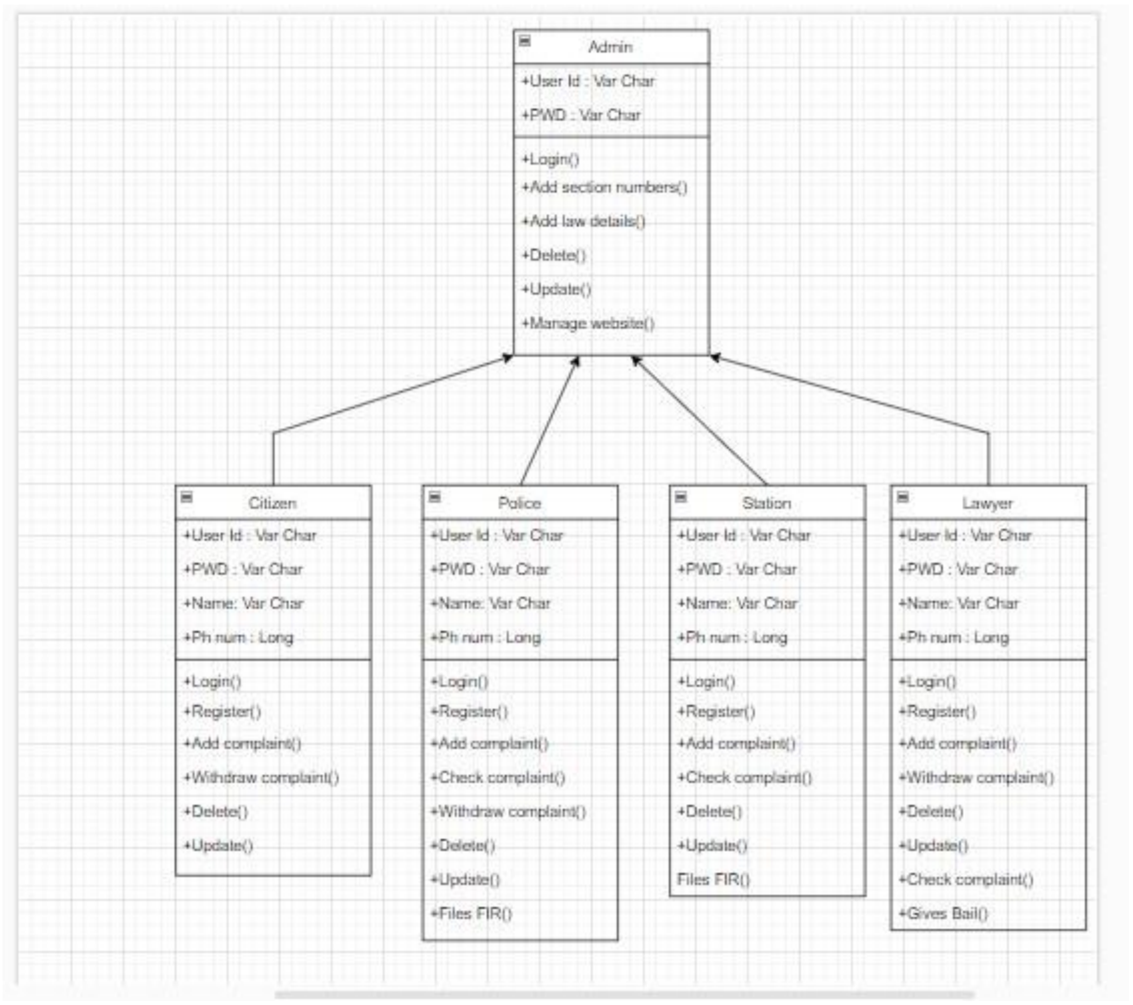
Property shared by all instances of a class. It lies in the second compartment in class rectangle.

- Operations

An execution of an action can be performed for any object of a class. It lies in the last compartment in class rectangle.

- Generalization/Specialization

It describes how one class is derived from another class. Derived class inherits the properties of its parent class.

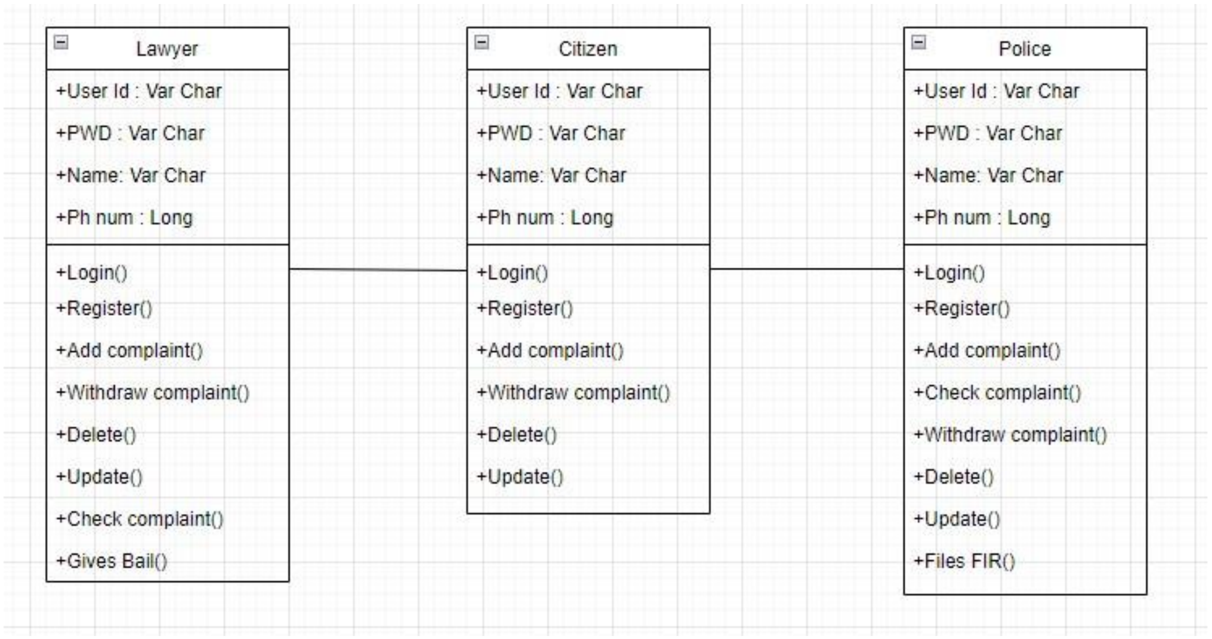


Relationships

Existing relationships in a system describe legitimate connections between the classes in that system.

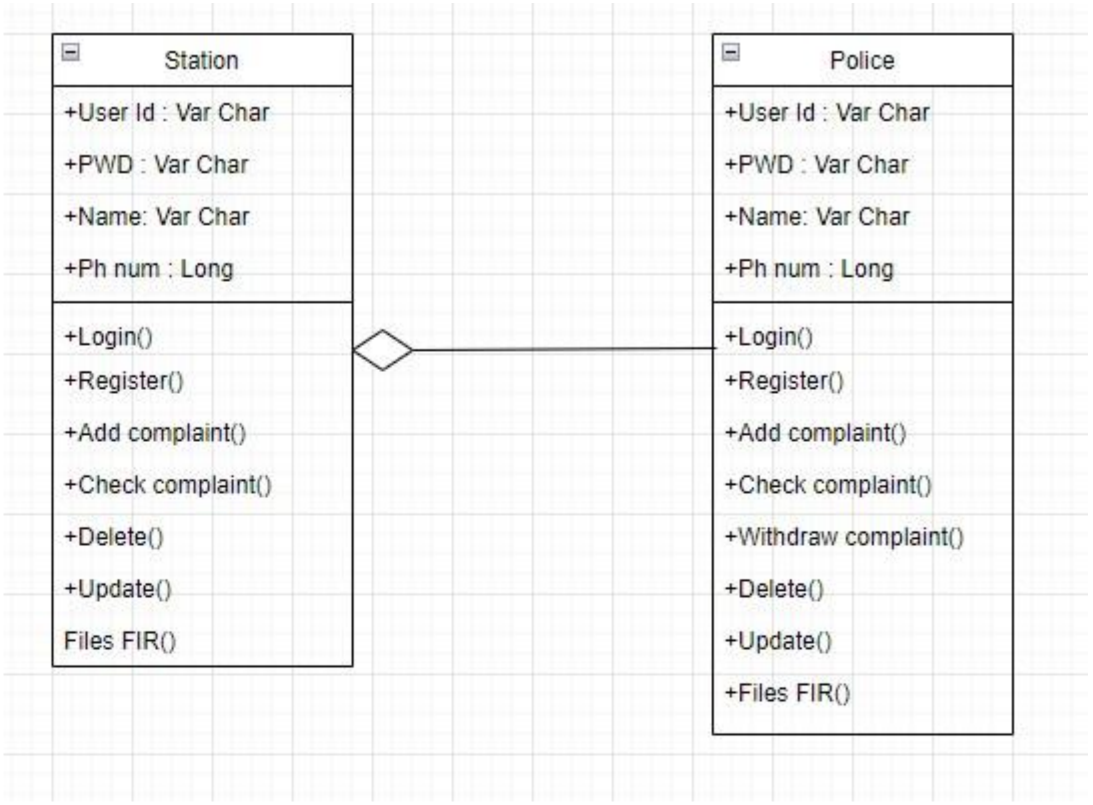
- Association

It is an instance level relationship[i] that allows exchanging messages among the objects of both ends of association. A simple straight line connecting two class boxes represent an association. We can give a name to association and also at the both end we may indicate role names and multiplicity of the adjacent classes. Association may be uni-directional.



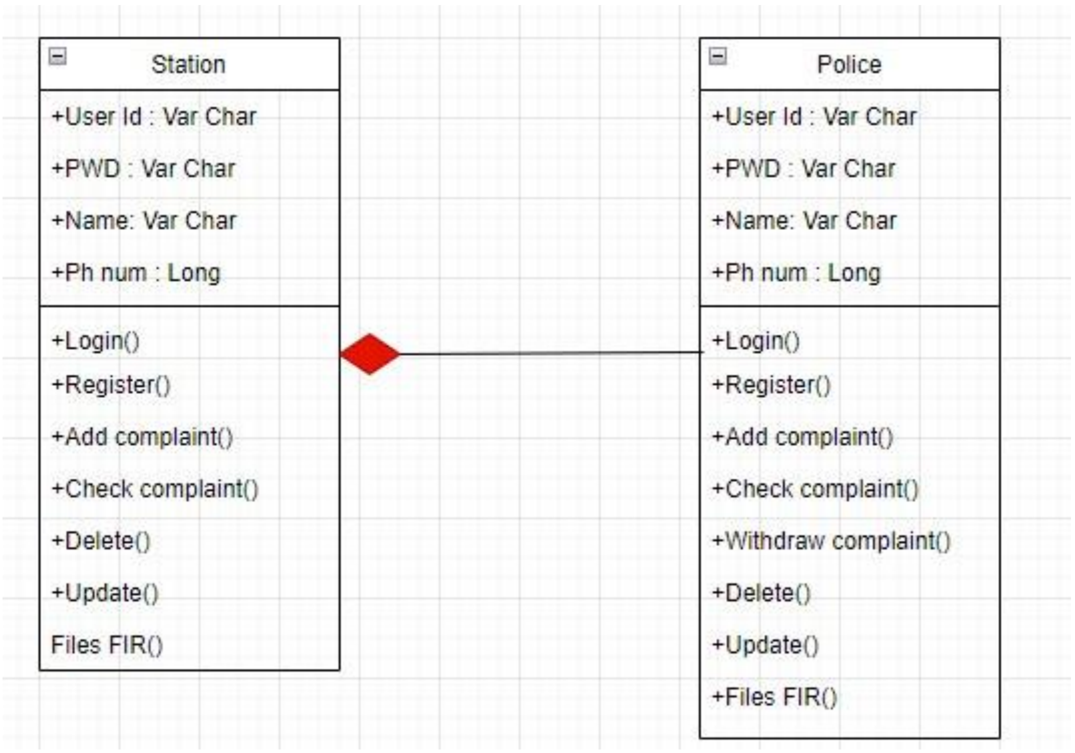
- Aggregation

It is a special form of association which describes a part-whole[i] relationship between a pair of classes. It means, in a relationship, when a class holds some instances of related class, then that relationship can be designed as an aggregation.



- Composition [i]

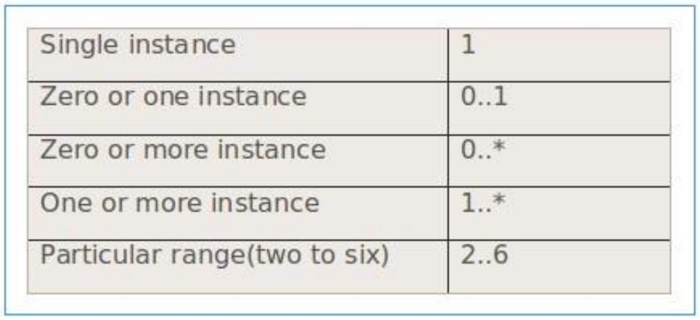
It is a strong from of aggregation which describes that whole is completely owns its part. Life cycle of the part depends on the whole.



- Multiplicity

It describes how many numbers of instances of one class is related to the number of instances of another class in an association.

Notation for different types of multiplicity:



Single instance	1
Zero or one instance	0..1
Zero or more instance	0..*
One or more instance	1..*
Particular range(two to six)	2..6

MODELING UML SEQUENCE DIAGRAMS:

Sequence diagram

It represents the behavioral aspects of a system. Sequence diagram shows the interactions between the objects by means of passing messages from one object to another with respect to time in a system.

Elements in sequence diagram

Sequence diagram contains the objects of a system and their life-line bar and the messages passing between them.

Object

Objects appear at the top portion of sequence diagram. Object is shown in a rectangle box. Name of object precedes a colon ':' and the class name, from which the object is instantiated. The whole string is underlined and appears in a rectangle box. Also, we may use only class name or only instance name.

Objects which are created at the time of execution of use case and are involved in message passing , are appear in diagram, at the point of their creation.

Life-line bar

A down-ward vertical line from object-box is shown as the life-line of the object. A rectangle bar on life-line indicates that it is active at that point of time.

Messages

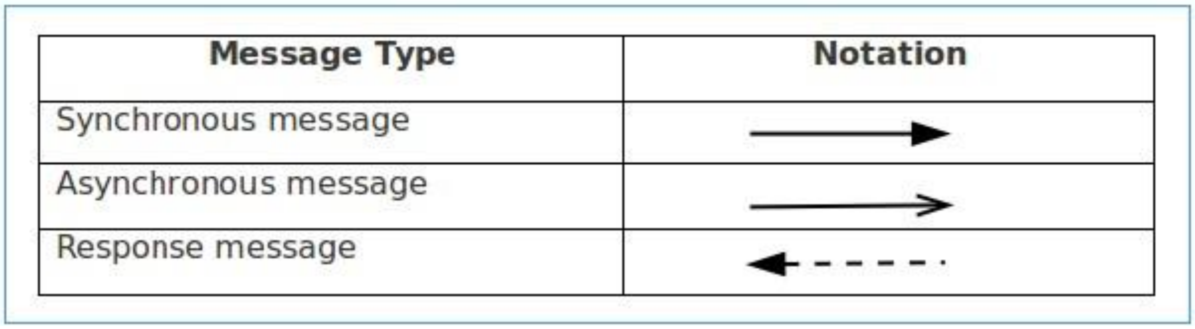
Messages are shown as an arrow from the life-line of sender object to the life-line of receiver object and labeled with the message name. Chronological order of the messages passing throughout the objects' life-line show the sequence in which they occur . There may exist some different types of messages :

Synchronous messages:Receiver start processing the message after receiving it and sender needs to wait until it is made. A straight arrow with close and fill arrow-head from sender life-line bar to receiver end, represent a synchronous message.

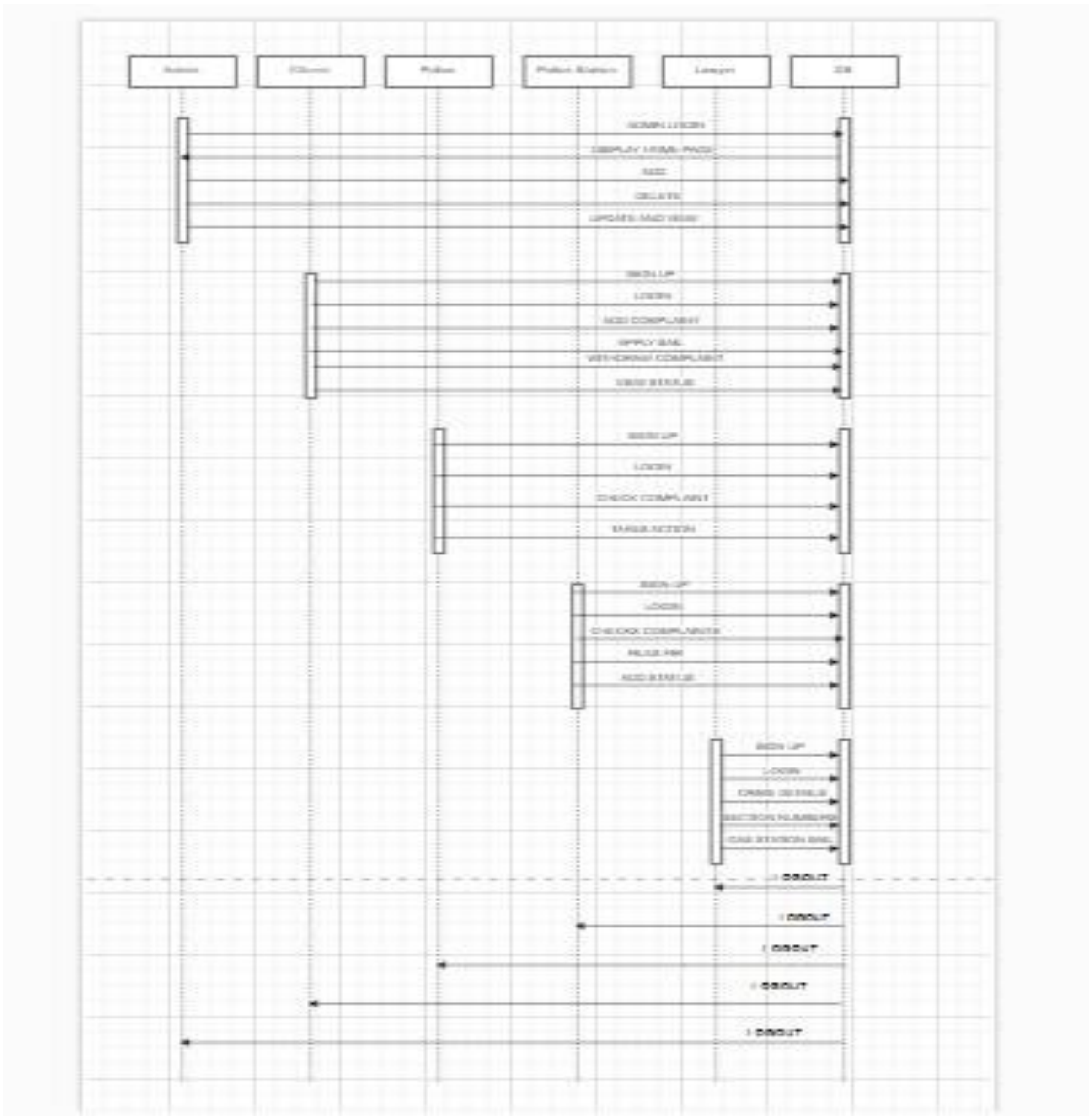
Asynchronous messages:For asynchronous message sender needs not to wait for the receiver to process the message. A function call that creates thread can be represented as an asynchronous message in sequence diagram. A straight arrow with open arrow-head from sender life-line bar to receiver end, represent an asynchronous message.

Return message:For a function call when we need to return a value to the object, from which it was called, then we use return message. But, it is optional, and we are using it when we are going to model our system in much detail. A dashed arrow with open arrowhead from sender life-line bar to receiver end, represent that message.

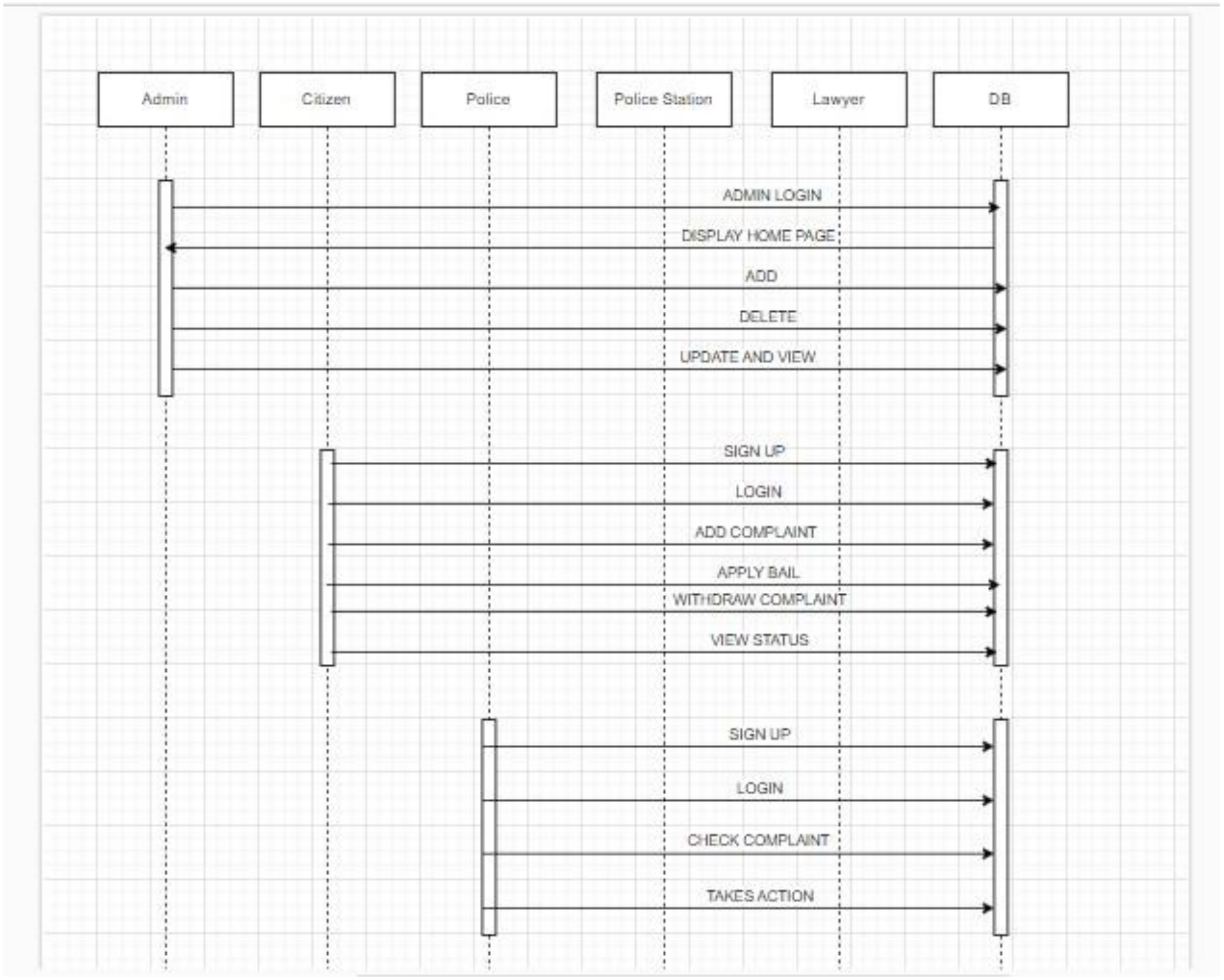
Response message:One object can send a message to self. We use this message when we need to show the interaction between the same object.

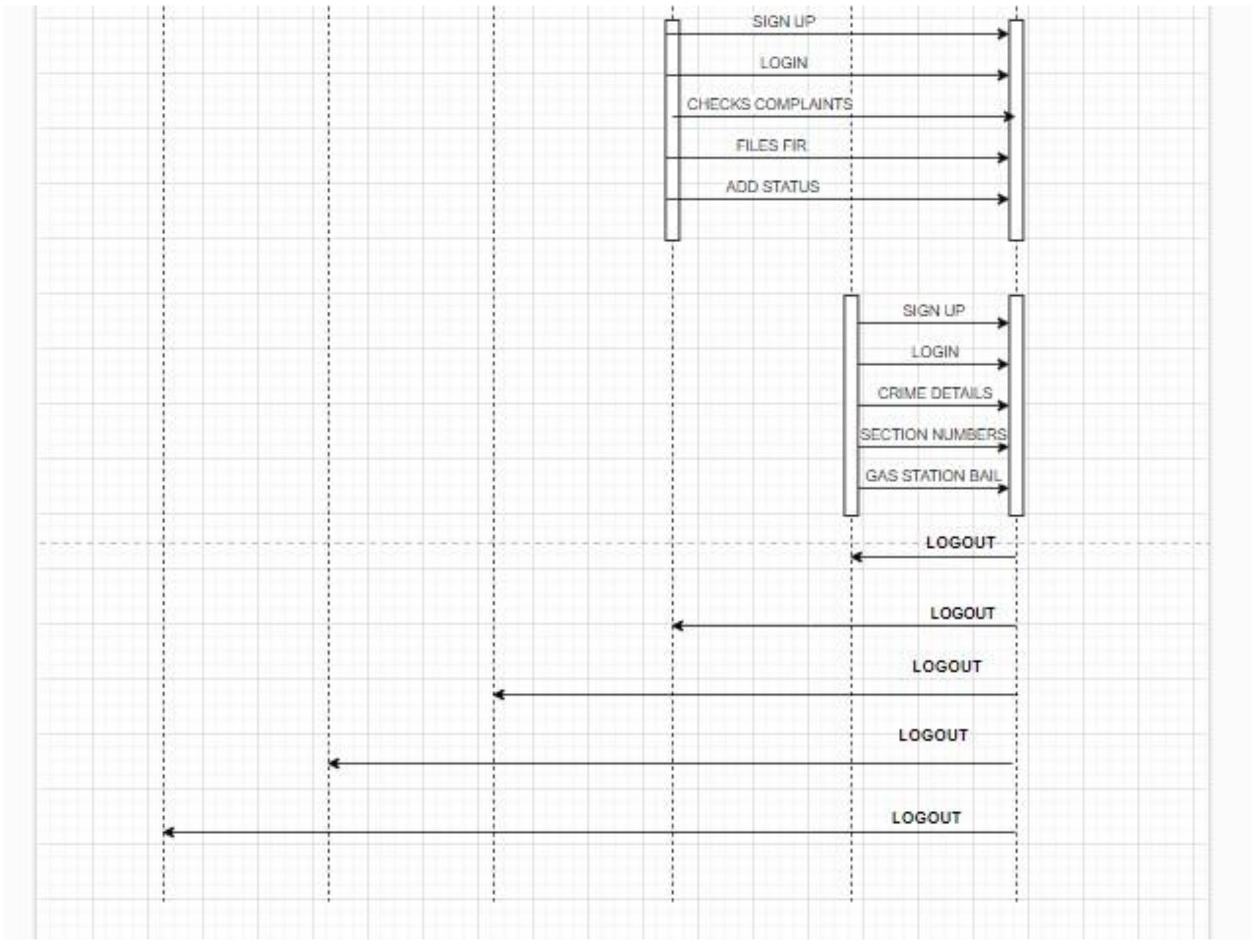


Message Type	Notation
Synchronous message	➞
Asynchronous message	→
Response message	⇠


Admin
Citizen
Police
Police Station
Lawyer
DB
ADMIN LOGIN
DISPLAY HOME PAGE
ADD
DELETE
UPDATE AND VIEW
SIGN UP
LOGIN
ADD COMPLAINT
APPLY BAIL
WITHDRAW COMPLAINT
VIEW STATUS
SIGN UP
LOGIN
CHECK COMPLAINT
TAKE ACTION
SIGN UP
LOGIN
CHECK COMPLAINTS
FILE FIR
ADD STATUS
SIGN UP
LOGIN
CASE DETAILS
SECTION NUMBERS
CASE STATION BAIL
LOGOUT
LOGOUT
LOGOUT
LOGOUT
LOGOUT

,


Admin
Citizen
Police
Police Station
Lawyer
DB
ADMIN LOGIN
DISPLAY HOME PAGE
ADD
DELETE
UPDATE AND VIEW
SIGN UP
LOGIN
ADD COMPLAINT
APPLY BAIL
WITHDRAW COMPLAINT
VIEW STATUS
SIGN UP
LOGIN
CHECK COMPLAINT
TAKES ACTION



E-R MODELING FROM THE PROBLEM STATEMENTS:

- This ER (Entity Relationship) Diagram represents the model of a Crime Record Management System Entity. The entity-relationship diagram of the Crime Record Management System shows all the visual instruments of database tables and the relations between Complaints, Charge Sheets, Crimes, Criminals, etc.
- Here are the main entities of the Crime Record Management System Crime, Complaints, FIR, Charge Sheet, Prisoner, and Criminals.

Crime Record Management System entities and their attributes:

- Crime Entity: Attributes of Crime are crime_id, crime_criminal_id, crime_name, crime_type, crime_description

- Complaints Entity: Attributes of Complaints are complain_id, complain_name, complain_type, complain_description

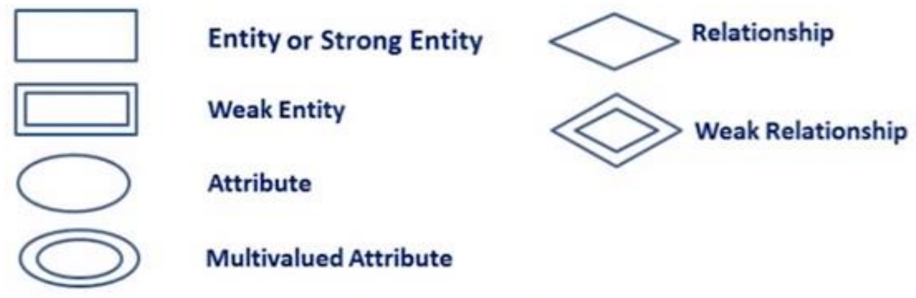
- FIR Entity: Attributes of FIR are FIR_id, FIR_name, FIR_type, FIR_description

- Charge Sheet Entity: Attributes of Charge Sheet are charger_sheet_id, charger_sheet__fine, charger_sheet_type, charger_sheet_description

- Presioner Entity: Attributes of Presioner are prisoner_id, prisoner_crime_id, prisoner_name, prisoner_mobile, prisoner_email, prisoner_username, prisoner_password, prisoner_address

- Criminals Entity: Attributes of Criminals are criminal_id, criminal_crime_id, criminal_name, criminal_mobile, criminal_email, criminal_username, criminal_password, criminal_address

- **Rectangles:** This Entity Relationship Diagram symbol represents entity types
- **Ellipses:** Symbols represent attributes
- **Diamonds:** This symbol represents relationship types
- **Lines:** It links attributes to entity types and entity types with other relationship types
- **Primary key:** attributes are underlined
- **Double Ellipses:** Represent multi-valued attributes

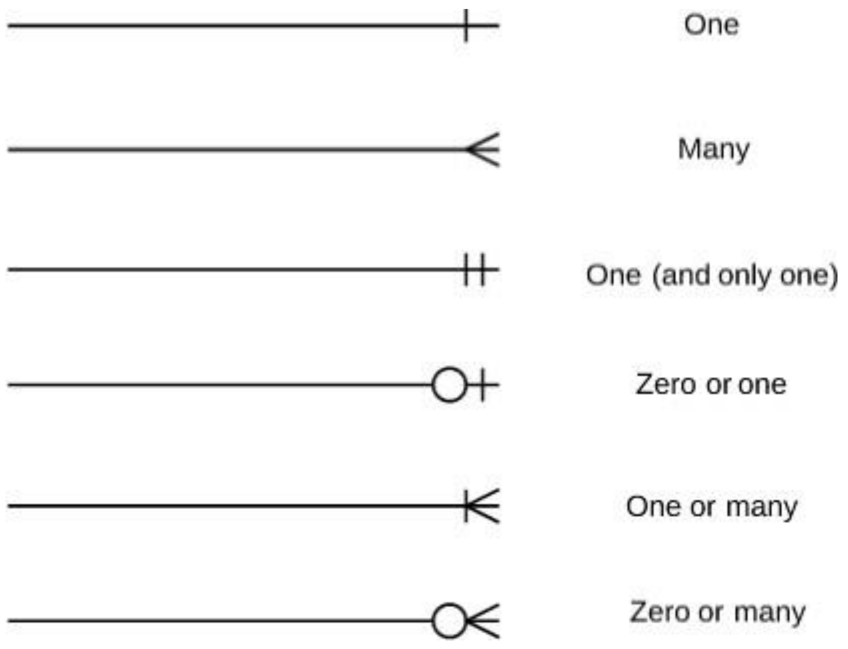


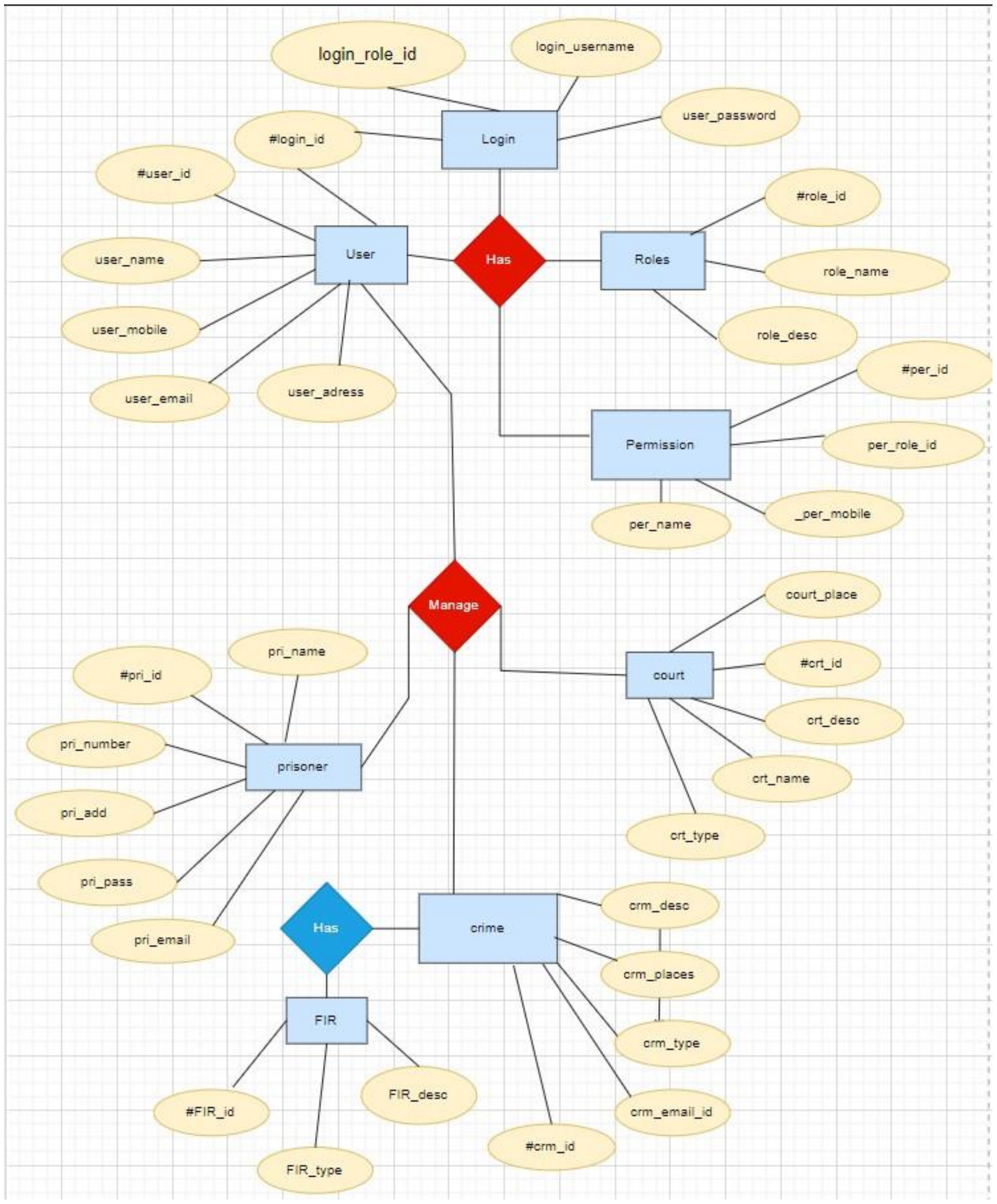
ER Diagram Symbols

Cardinality and ordinality

Cardinality refers to the maximum number of times an instance in one entity can relate to instances of another entity. Ordinality, on the other hand, is the minimum number of times an instance in one entity can be associated with an instance in the related entity.

Cardinality and ordinality are shown by the styling of a line and its endpoint, according to the chosen notation style.





STATECHART MODELING:

The name of the diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system. The states are specific to a component/object of a system.
A Statechart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events.

Activity diagram explained in the next chapter, is a special kind of a Statechart diagram. As Statechart diagram defines the states, it is used to model the lifetime of an object.

Purpose of Statechart Diagrams

Statechart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

Statechart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of Statechart diagram is to model lifetime of an object from creation to termination.

Statechart diagrams are also used for forward and reverse engineering of a system. However, the main purpose is to model the reactive system.

Following are the main purposes of using Statechart diagrams −

- To model the dynamic aspect of a system.
- To model the life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model the states of an object.

How to Draw a Statechart Diagram?

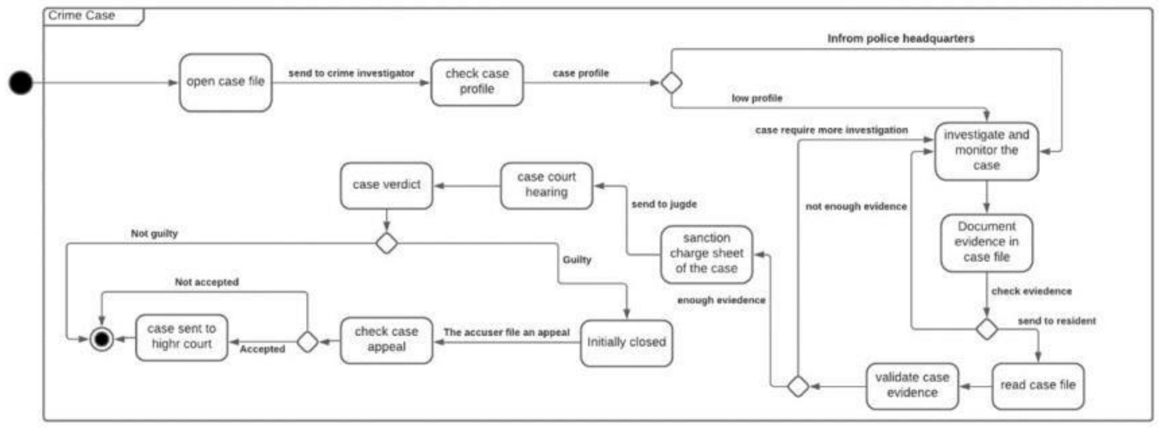
Statechart diagram is used to describe the states of different objects in its life cycle. Emphasis is placed on the state changes upon some internal or external events. These states of objects are important to analyze and implement them accurately.

Statechart diagrams are very important for describing the states. States can be identified as the condition of objects when a particular event occurs.

Before drawing a Statechart diagram we should clarify the following points −

- Identify the important objects to be analyzed.
- Identify the states.
- Identify the events.

Following is an example of a Statechart diagram where the state of Order object is analyzed



StateChart diagrams represent the behaviour of an object. There are different states of an object within the system.

The statechart diagram here illustrates the state of the Crime Case File, which is the main object that has various states in the system. The statechart diagram is initiated by opening a case file. After that, “check case profile” state is triggered when the case file is sent to the crime investigator. If the case is high profile, the crime investigator informs police headquarter. Otherwise, when the case is low profile, the crime investigator starts the investigating process. During the investigation, the evidence is documented in the crime case file. The following state of the Crime Case File is “Reading Case File” by the resident state attorney.

The next state is validating case evidence, which triggers a decision of requesting more investigation or the evidence is enough, then, sanction charge sheet of the case. Hence, the “Case Court Hearing” state is triggered by sending the case file to the judge, which results in deciding the verdict. If the verdict is not guilty, the state of the crime case file is terminated. However, if the verdict is guilty, the case is initially closed. The “Check Case Appeal” is triggered when the accuser files an appeal. If the appeal is not accepted by the judge, the case would be finally closed. Otherwise, the case would be sent to the higher court to give the final verdict and close the case. A decision that is determined by crime investigator is whether to send the case file to the resident state attorney, or not, and that depends on whether there is enough evidence or not. If the evidence is not enough, the crime investigator shall continue investigations.

Where to Use Statechart Diagrams?

From the above discussion, we can define the practical applications of a Statechart diagram. Statechart diagrams are used to model the dynamic aspect of a system like other four diagrams discussed in this tutorial. However, it has some distinguishing characteristics for modeling the dynamic nature.

Statechart diagram defines the states of a component and these state changes are dynamic in nature. Its specific purpose is to define the state changes triggered by events. Events are internal or external factors influencing the system.

Statechart diagrams are used to model the states and also the events operating on the system. When implementing a system, it is very important to clarify different states of an object during its life time and Statechart diagrams are used for this purpose. When these states and events are identified, they are used to model it and these models are used during the implementation of the system.

If we look into the practical implementation of Statechart diagram, then it is mainly used to analyze the object states influenced by events. This analysis is helpful to understand the system behavior during its execution.

The main usage can be described as −

- To model the object states of a system.
- To model the reactive system. Reactive system consists of reactive objects.
- To identify the events responsible for state changes.
- Forward and reverse engineering.

ACTIVITY MODELING:

Activity diagrams fall under the category of behavioural diagrams in Unified Modeling Language. It is a high level diagram used to visually represent the flow of control in a system. It has similarities with traditional flow charts. However, it is more powerful than a simple flow chart since it can represent various other concepts like concurrent activities, their joining, and so on.

Activity diagrams, however, cannot depict the message passing among related objects. As such, it can't be directly translated into code. These kind of diagrams are suitable for confirming the logic to be implemented with the business users. These diagrams are typically used when the business logic is complex. In simple scenarios it can be avoided entirely.

Components of an Activity Diagram

Below we describe the building blocks of an activity diagram.

Activity

An activity denotes a particular action taken in the logical flow of control. This could simply be invocation of a mathematical function, alter an object's properties and so on [x]. An activity is represented with a rounded rectangle, as shown in table-01. A label inside the rectangle identifies the corresponding activity.

There are two special type of activity nodes: initial and final. They are represented with a filled circle, and a filled in circle with a border respectively (table-01). Initial node represents the starting point of a flow in an activity diagram. There could be multiple initial nodes, which means that invoking that particular activity diagram would initiate multiple flows.

A final node represents the end point of all activities. Like an initial node, there could be multiple final nodes. Any transition reaching a final node would stop all activities.

Flow

A flow (also termed as edge, or transition) is represented with a directed arrow. This is used to depict transfer of control from one activity to another, or to other types of components, as we will see below. A flow is often accompanied with a label, called the guard condition, indicating the necessary condition for the transition to happen. The syntax to depict it is [guard condition].

Decision

A decision node, represented with a diamond, is a point where a single flow enters and two or more flows leave. The control flow can follow only one of the outgoing paths. The outgoing edges often have guard conditions indicating true-false or if-then-else conditions. However, they can be omitted in obvious cases. The input edge could also have guard conditions. Alternately, a note can be attached to the decision node indicating the condition to be tested.

Merge

This is represented with a diamond shape, with two or more flows entering, and a single flow leaving out. A merge node represents the point where at least a single control should reach before further processing could continue.

Fork

Fork is a point where parallel activities begin. For example, when a student has been registered with a college, he can in parallel apply for student ID card and library card. A fork is graphically depicted with a black bar, with a single flow entering and multiple flows leaving out.

Join

A join is depicted with a black bar, with multiple input flows, but a single output flow. Physically it represents the synchronization of all concurrent activities. Unlike a merge, in case of a join all of the incoming controls must be completed before any further progress could be made. For example, a sales order is closed only when the customer has receive the product, and the sales company has received it's payment.

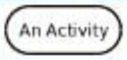
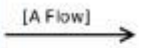
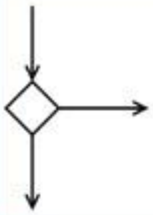
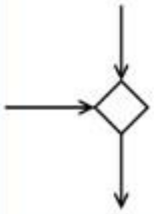
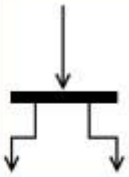
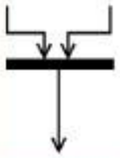
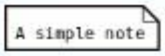
Note

UML allows attaching a note to different components of a diagram to present some textual information. The information could simply be a comment or may be some constraint. A note can be attached to a decision point, for example, to indicate the branching criteria.

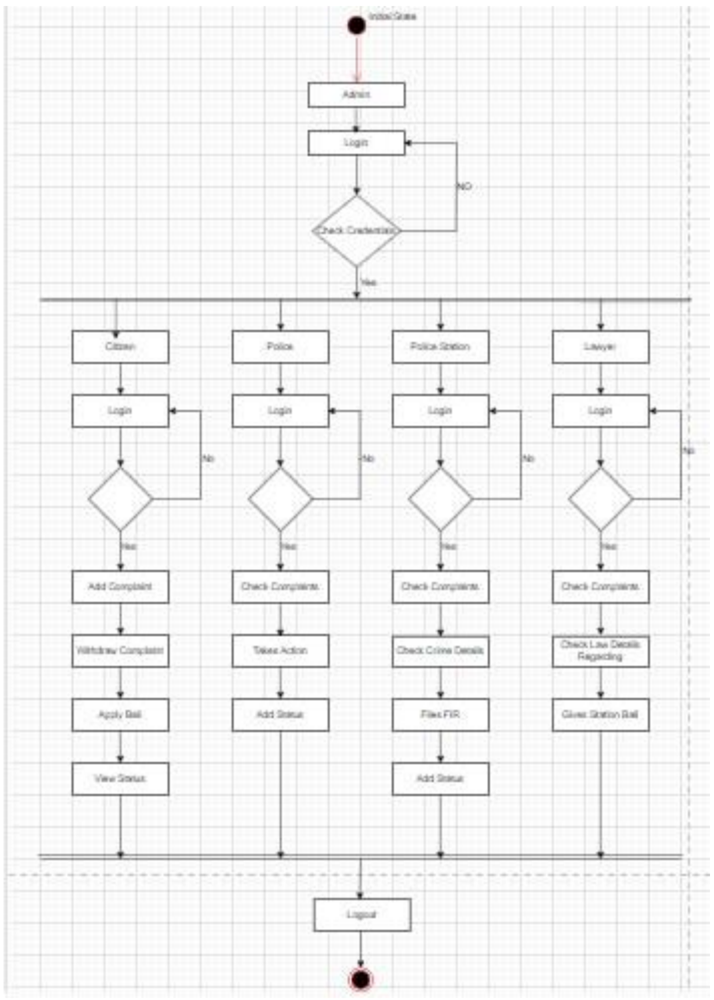
Partition

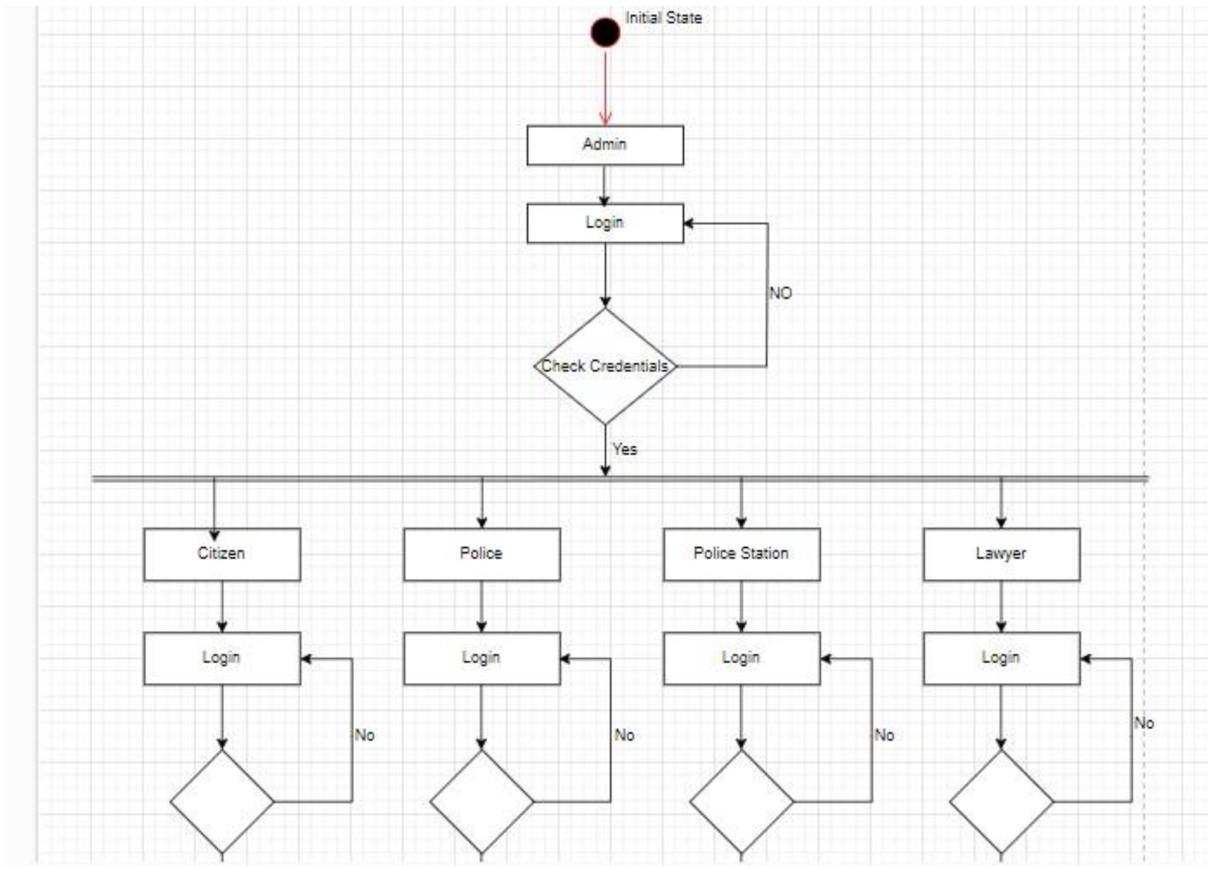
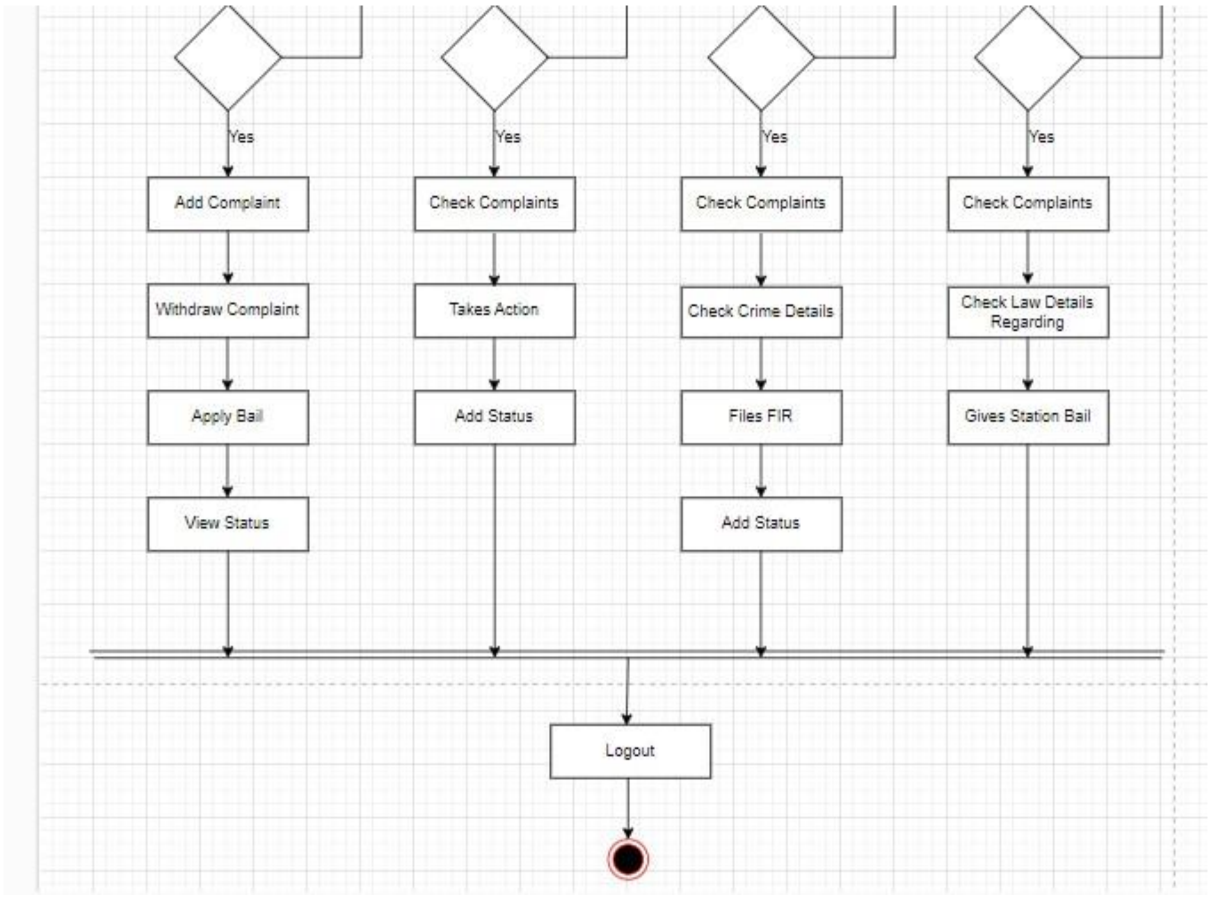
Different components of an activity diagram can be logically grouped into different areas, called partitions or swimlanes. They often correspond to different units of an organization or different actors. The drawing area can be partitioned into multiple compartments using vertical (or horizontal) parallel lines. Partitions in an activity diagram are not mandatory.

The following table shows commonly used components with a typical activity diagram.

Component	Graphical Notation
Activity	
Flow	
Decision	
Merge	
Fork	
Join	
Note	

Activity Diagram:


Initial State
Admin
Login
NO
Check Credentials
Yes
Citizen
Police
Police Station
Lawyer
Login
Login
Login
Login
No
No
No
No
Yes
Yes
Yes
Yes
Add Complaint
Check Complaints
Check Complaints
Check Complaints
Withdraw Complaint
Takes Action
Check Crime Details
Check Law Details Regarding
Apply Bail
Add Status
Files FIR
Gives Station Bail
View Status
Add Status
Logout


Initial State
Admin
Login
NO
Check Credentials
Yes
Citizen
Police
Police Station
Lawyer
Login
Login
Login
Login
No
No
No
No

Yes
Yes
Yes
Yes
Add Complaint
Check Complaints
Check Complaints
Check Complaints
Withdraw Complaint
Takes Action
Check Crime Details
Check Law Details Regarding
Apply Bail
Add Status
Files FIR
Gives Station Bail
View Status
Add Status
Logout

MODELING DATA FLOW DIAGRAM

What is a data flow diagram?

· A data flow diagram (DFD) maps out the flow of information for any process or system.

· It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.

· Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled.

· They can be used to analyze an existing system or model a new one.

· Like all the best diagrams and charts, a DFD can often visually "say" things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO.

· That's why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems

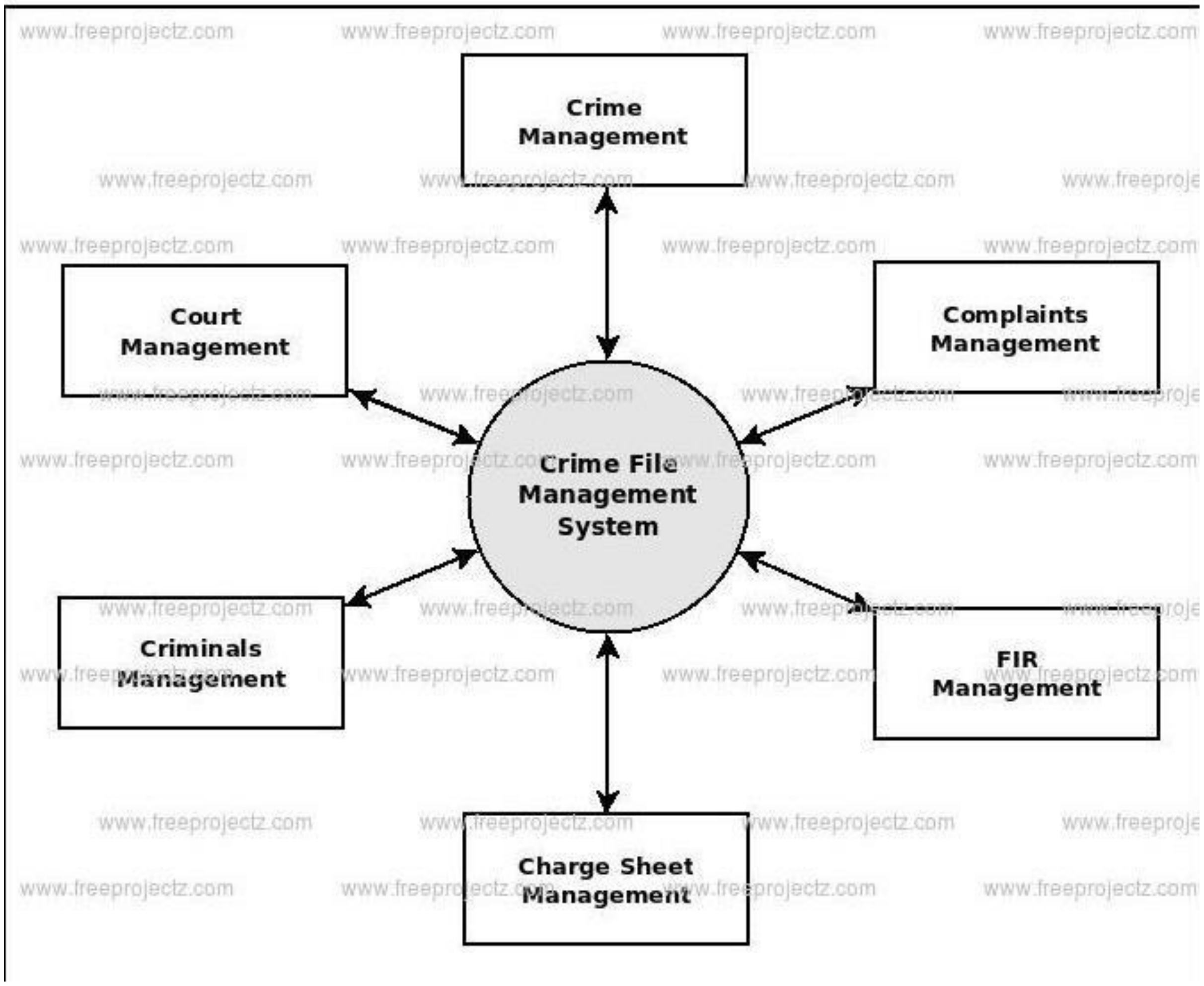
DFD levels and layers: From context diagrams to pseudocode

A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece.

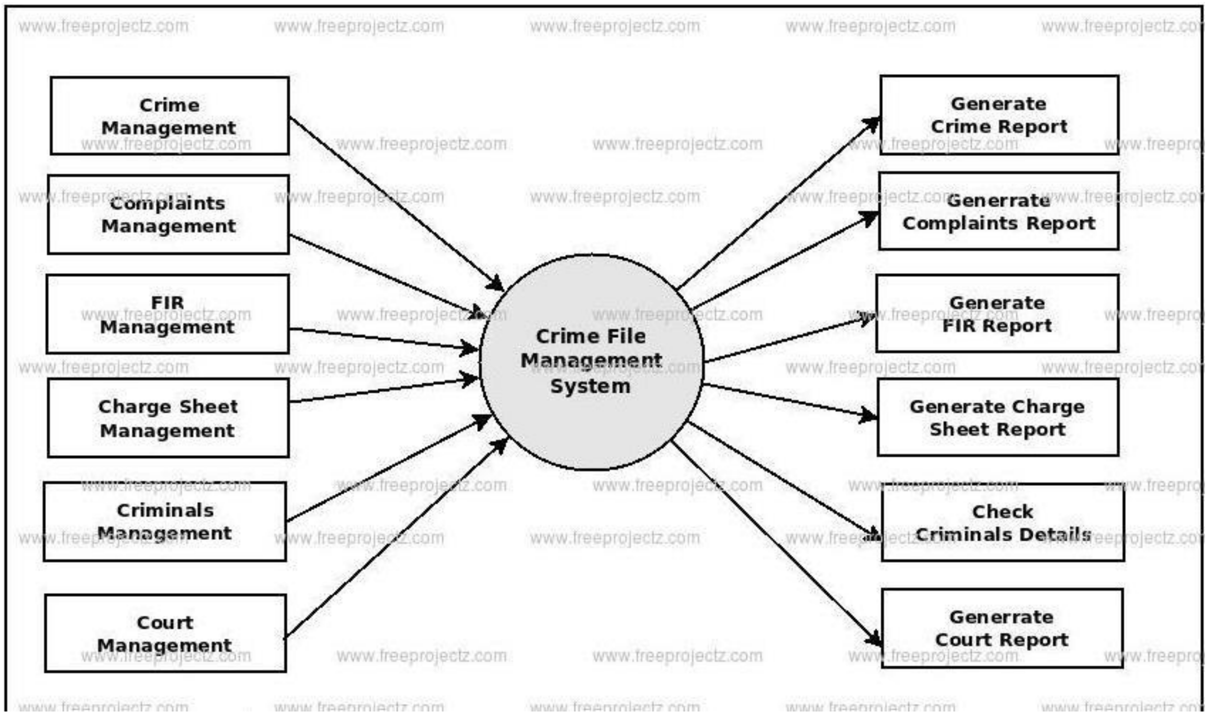
DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond.

The necessary level of detail depends on the scope of what you are trying to accomplish.

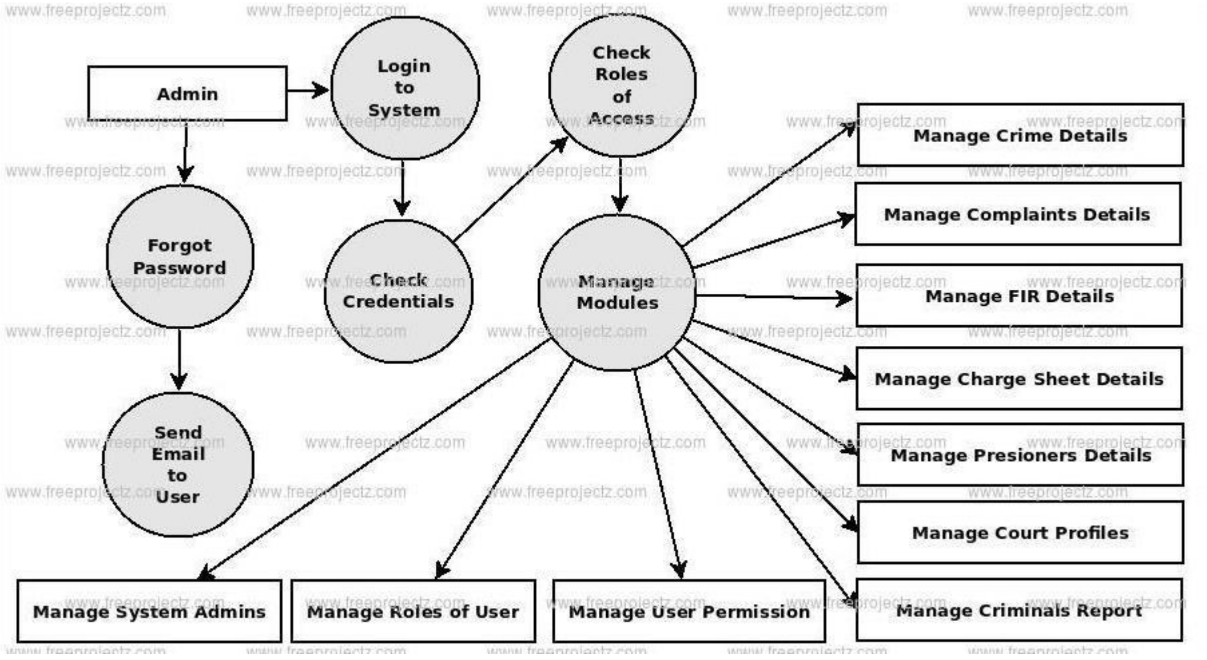
· DFD Level 0 is also called a Context Diagram. It's a basic overview of the whole system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.



· DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into its subprocesses.



· DFD Level 2 then goes one step deeper into parts of Level 1. It may require more text to reach the necessary level of detail about the system's functioning.



· Progression to Levels 3, 4 and beyond is possible, but going beyond Level 3 is uncommon. Doing so can create complexity that makes it difficult to communicate, compare or model effectively.

Using DFD layers, the cascading levels can be nested directly in the diagram, providing a cleaner look with easy access to the deeper dive.

By becoming sufficiently detailed in the DFD, developers and designers can use it to write pseudocode, which is a combination of English and the coding language.

Pseudocode facilitates the development of the actual code

ESTIMATION OF TEST COVERAGE METRICS AND STRUCTURAL COMPLEXITY:

Control Flow Graph

A control flow graph (CFG) is a directed graph where the nodes represent different instructions of a program, and the edges define the sequence of execution of such instructions.

Graph-theoretic analysis is used to determine the complexity of a program module . Cyclomatic complexity metric, provides an upper bound for the number of linearly independent paths that could exist through a given program module. Complexity of a module increases as the number of such paths in the module increase. Thus, if Cyclomatic complexity of any program module is 7, there could be up to seven linearly independent paths in the module. For a complete testing, each of those possible paths should be tested.

DESIGNING TEST SUITES:

Test Reports, Test Plan:

Software testing is a process of validating and verifying that a software program /application/product meets:

- The business and technical requirements that guided its design and development.
- Works as expected can be implemented with the same characteristics.

Purpose:

One of the main goals of testing is to find software flaws so that faults can be found and fixed. Software testing entails both the inspection and execution of code in a variety of contexts and scenarios.

- Examining the characteristics of code, such as whether it does what it is supposed to do and what it needs to do, is also part of the process. Information gleaned from software testing can be utilised to improve the software development process.

Testing Methods:

White Box Testing:

White box testing is done when the tester has access to the internal data structures and algorithms including the code that implements it.

Black Box Testing:

This type of software testing does not require knowledge of how the software functions.

- It is conducted without knowing how the product works.

Grey Box Testing:

The purpose of grey box testing is to understand internal data structures and algorithms when designing test cases, but it is to test the system at the user, or black box, level.

- In addition to grey box testing, reverse engineering can be used to determine, for instance, boundary values or error messages.

Testing Levels:

The software development process often groups tests by where they are placed within it, or by the specificity of the test Unit Testing:

- In the object-oriented environment, unit tests usually cover constructors and destructors as a minimum. In addition to functions, unit tests also usually test the behavior of classes.

Integration testing:

Integration testing involves verifying the interfaces of components

within the software

System testing:

The purpose of system testing is to verify that a system is fully integrated and meets its requirements.

System Integration Testing:

Testing the integration of a system with external or third-party systems in the requirements ensures that it is able to access them.

Test Report 1:

Project Name: CRIME RECORDS MANAGEMENT SYSTEM

Form Name: Login

Unit Name: User-id, Password

Test Result: After entering two fields the user successfully logs into the system

Test Plan 1:

Unit id : Login

Test Case id : User-

id

Test Type : Unit

Testing Form Name :

Login

Base Table : Admin/Citizen/POLICE

Purpose

Test data:

Serial No.	Input Specification	Output Specification
1	**User id:** Valid Input Invalid input	Navigates to the respective home pages. It will ask to enter correct values again.
2	**Password:** Valid Input Invalid input	Navigates to the respective home page. It will ask to enter correct values again.

Test Process:

Login form will be used for allowing the correct user to use the software. Every person will be given a user id and password. After successful login the user can use the software as per the privileges given to him. The user id will be entered in the textbox given for user id.

The password will be entered in the textbox given for the password.

Test completion criteria:

When expected results match the actual results after performing the test, the test is considered to be complete.

CONCLUSION

- The **Crime Records Management System** is a web-based application for primarily providing training to the employees who provide customized solutions to meet organizational needs.

- This application software has been computed successfully and was also tested successfully by taking "test cases". It is user friendly, and has required options, which can be utilized by the user to perform the desired operations.

- The software is developed using ASP.Net as front end and SQL as back end in the Windows environment. The goals that are achieved by the software are:

  ✓ Instant access.

  ✓ Improved productivity.

  ✓ Optimum utilization of resources.

  ✓ Efficient management of records.

  ✓ Simplification of the operations.

  ✓ Less processing time and getting required information.

  ✓ User friendly.