Problem Statement

Problem Statement:

In today's world, social media platforms such as Twitter have become a ubiquitous source of information for people all over the world. Millions of people share their opinions, thoughts, and feelings on various topics using Twitter. This vast amount of user-generated data presents a great opportunity for researchers and analysts to gain valuable insights into public opinion on different issues.

However, analyzing this massive amount of data manually is not feasible, as it requires a significant amount of time and effort. Therefore, there is a need for automated systems that can collect and process Twitter data to identify trends and patterns in public sentiment on various topics.

The aim of this project is to develop a software application that can collect Twitter data related to a specific topic, perform sentiment analysis, and predict future trends based on the emotions of the tweets. The project will use natural language processing (NLP) techniques and machine learning algorithms to analyze the data.

Use Cases:

The proposed software application can be used in various domains and industries. For instance, it can be used by marketing agencies to understand consumer behavior and preferences, by political analysts to analyze public opinion on political issues, by healthcare organizations to monitor public sentiment on health-related topics, and by news agencies to analyze public reaction to breaking news.

Applications:

The application has several potential applications, including:

1. Political Analysis: The software application can be used to monitor public opinion on political issues such as elections, policy changes, and governance. It can help political parties and candidates to develop strategies to engage with their constituents and gain support.

2. Marketing Analysis: The application can help companies to analyze consumer behavior and preferences, enabling them to develop targeted marketing campaigns that resonate with their audience.

3. Social Issue Analysis: The software application can be used to analyze public sentiment on social issues such as climate change, gun control, and gender equality. This information can be used by advocacy groups to develop effective campaigns and to influence public policy.

Challenges:

The project presents several challenges, including:

1. Data Collection: Collecting relevant data from Twitter can be challenging as it requires filtering out irrelevant data and ensuring that the collected data is representative of the entire population.

2. Sentiment Analysis: Sentiment analysis is a complex task that requires understanding the nuances of language, sarcasm, and context. Developing an accurate sentiment analysis model can be challenging, especially when dealing with large amounts of data.

3. Prediction: Predicting future trends based on sentiment analysis is a difficult task, as it requires identifying patterns and correlations in the data. Developing an accurate predictive model requires expertise in data analysis and machine learning.

Why this should be done:

Analyzing public sentiment on different issues is critical in today's world, as it can provide valuable insights into the opinions and preferences of people. It can be used to develop targeted marketing campaigns, to influence public policy, and to understand consumer behavior. The proposed software application can automate the process of data collection, sentiment analysis, and trend prediction, saving time and effort for researchers and analysts. The project has significant potential to provide useful insights that can be applied in various domains and industries.

In summary, this project aims to develop a software application that can collect Twitter data, perform sentiment analysis, and predict future trends based on public sentiment. The project has several potential applications and presents significant challenges, but has the potential to provide valuable insights that can be applied in various domains and industries.

Steps we should take:

1. Define the Problem and Scope: The first step in solving any problem is to clearly define the problem and scope of the project. We need to define the topic that we will be analyzing, the timeframe for data collection, and the specific sentiment categories that we will be analyzing.

2. Collect Data: Once we have defined the problem and scope of the project, we need to collect data from Twitter. There are various methods for collecting Twitter data, such as using Twitter's API, third-party tools, or web scraping. We need to ensure that the data collected is representative of the entire population and is relevant to the topic of interest.

3. Pre-process Data: The collected data needs to be pre-processed to remove any irrelevant information and to prepare the data for sentiment analysis. This involves removing URLs, special characters, and stop words. The data may also need to be tokenized and normalized to prepare it for analysis.

4. Perform Sentiment Analysis: Sentiment analysis involves classifying the tweets into different categories such as positive, negative, and neutral. There are various methods for performing sentiment analysis, such as using rule-based systems or machine learning algorithms. We need to select the appropriate method based on the specific requirements of the project.

5. Analyze Results: Once we have performed sentiment analysis, we need to analyze the results to identify trends and patterns in public sentiment. We can use data visualization tools such as graphs and charts to help identify these patterns.

6. Predict Future Trends: Based on the trends and patterns identified in the data analysis, we can predict future trends in public sentiment. This involves developing predictive models that can forecast future sentiment based on historical data.

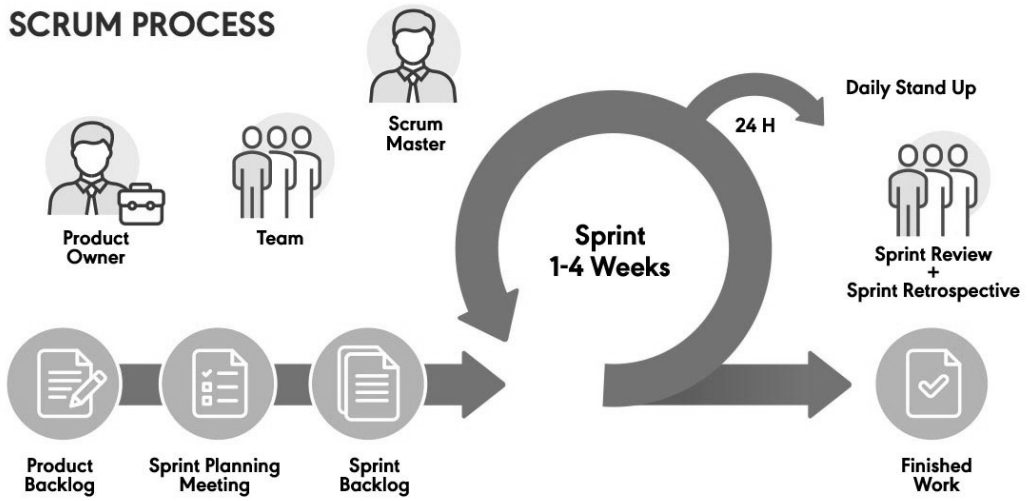
7. Evaluate and Refine: The final step is to evaluate the results and refine the analysis as necessary. We need to ensure that the analysis is accurate and reliable, and that the predictive models are effective in forecasting future trends.

Process Model (Scrum)

In our project, we will be using the Scrum process model to manage the development process. We have chosen this model because it is well-suited to our project, which involves analyzing Twitter data related to a specific topic and predicting trends based on the emotions of the tweets.

To implement the Scrum process model in our project, we will first define the Product Backlog, which will contain the features and requirements for the project. We will then break down the work into sprints, with each sprint lasting approximately 2-4 weeks. During each sprint, we will complete the work defined in the Sprint Backlog, which will be a subset of the Product Backlog. We will hold daily standup meetings to discuss progress and any obstacles that may be preventing progress.

At the end of each sprint, we will hold a Sprint Review to demonstrate the completed work to stakeholders and receive feedback. We will then hold a Sprint Retrospective to reflect on the sprint and identify areas for improvement. We will use this feedback to adjust the Product Backlog and plan the work for the next sprint.



We have chosen to use the Scrum process model for our project because it provides a number of advantages that are well-suited to our project needs.

One of the main advantages of the Scrum process model is that it is an Agile methodology, which means it is well-suited to projects that involve rapidly changing requirements and priorities. In our project, we will be analyzing Twitter data related to a specific topic and predicting trends based on the emotions of the tweets. This is a complex and dynamic task that requires flexibility and adaptability. Scrum provides a flexible framework that enables us to respond to changing requirements and priorities as they emerge.

Another advantage of the Scrum process model is that it is an iterative and incremental process. This means that we can deliver working software incrementally throughout the project, rather than waiting until the end to deliver a final product. This allows us to receive feedback from stakeholders early and often, which in turn enables us to make adjustments and improvements as we go. This approach helps us to ensure that we are delivering a high-quality software product that meets the needs of our stakeholders.

Scrum also provides a collaborative framework that encourages team members to work together and communicate effectively. This is important for our project, as we will be working with a team of developers, data analysts, and other stakeholders to analyze Twitter data and predict trends. Scrum provides a framework for collaboration, communication, and decision-making that enables us to work effectively as a team.

Overall, we are using the Scrum process model for our project because it is well-suited to our needs. It provides a flexible and iterative approach to software development that enables us to respond to changing requirements and priorities, deliver working software incrementally, and work collaboratively as a team. These advantages will help us to deliver a high-quality software product that meets the needs of our stakeholders.

Data Flow Diagram

Definition:

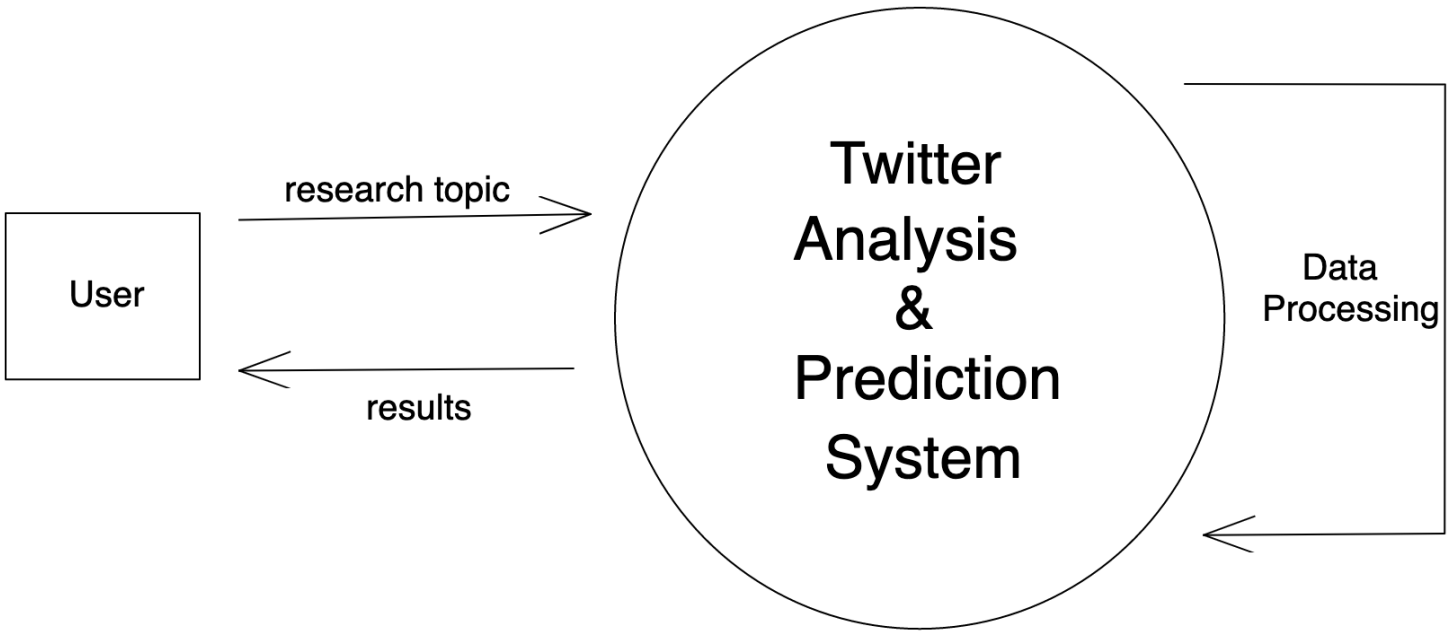
A data flow diagram (DFD) is a graphical representation of how data flows through a system. It is used to model a system's processes, data stores, and external entities, and to show how data moves through the system. A DFD is a powerful tool for analyzing, designing, and documenting complex systems.

DFDs can be divided into different levels, each representing a different level of abstraction.

Each level of a DFD provides a different level of detail about the system being modeled, and each level builds on the previous level. The highest-level DFD (Context Diagram) provides an overview of the system, while the lower-level DFDs provide more detail about specific processes or functions in the system.

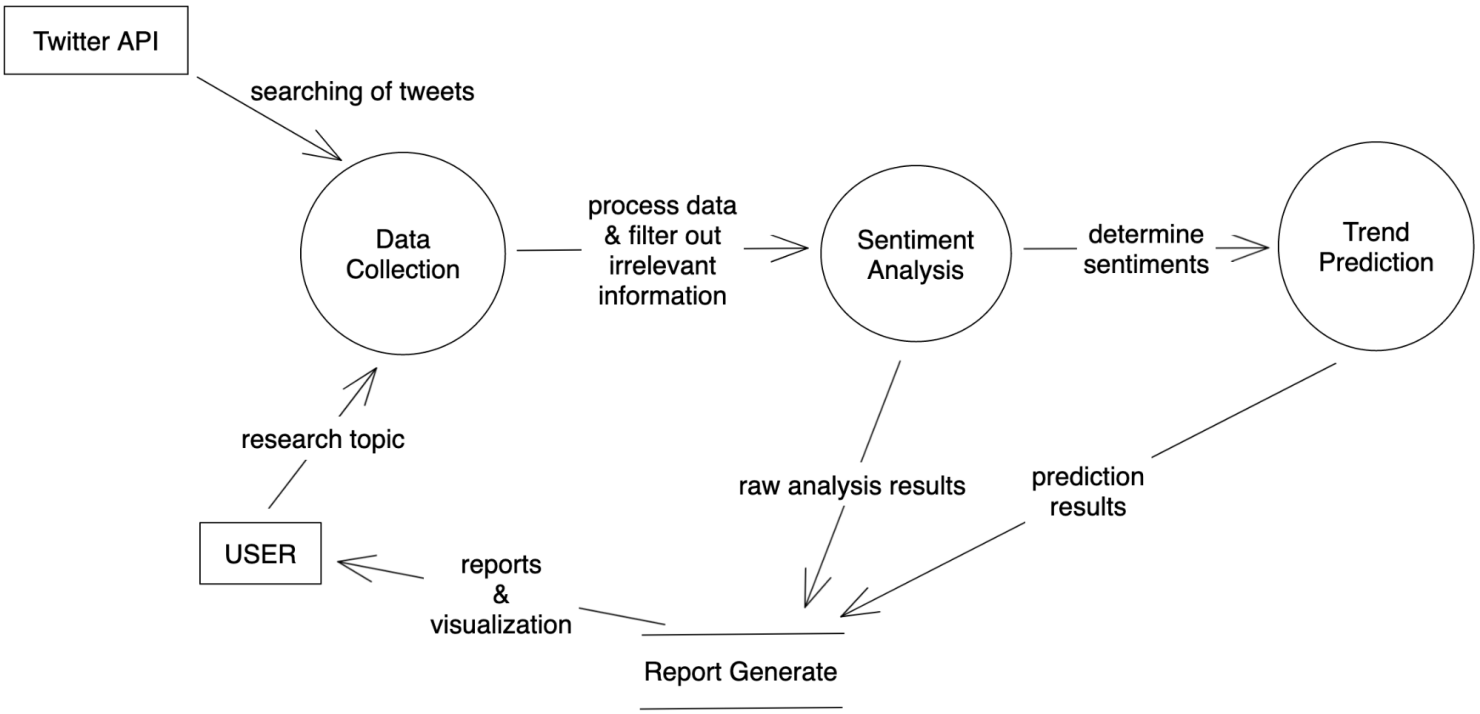
Context Level (Level 0):

Level 0 - Context Diagram: The Context Diagram is the highest-level DFD, and it provides an overview of the entire system. It represents the system as a single process or function, surrounded by external entities (sources and sinks of data), and shows the flow of data between the system and external entities.



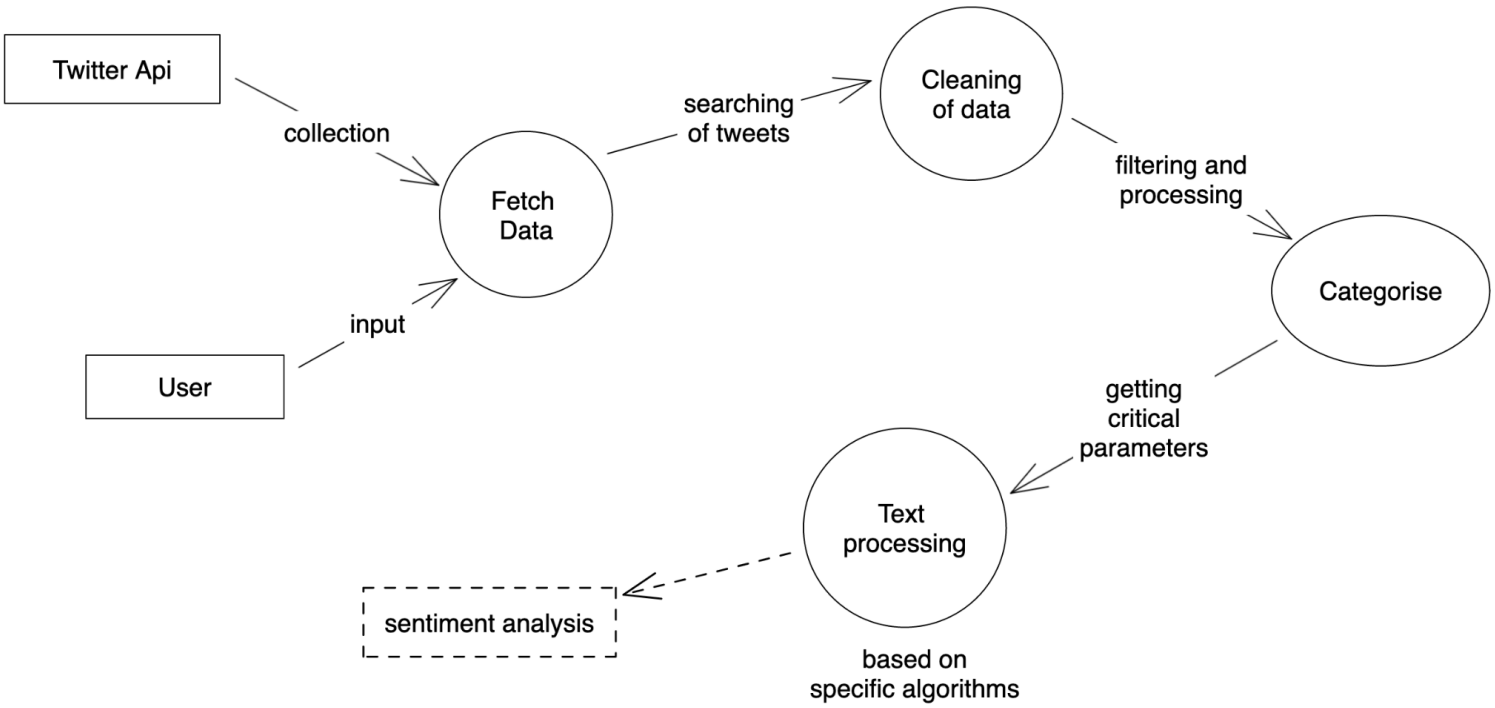
Level 1 (Top-level DFD):

Level 1 - Top-Level DFD: The Top-Level DFD shows the major processes or functions within the system and how they interact with each other. It represents the system as a set of interconnected processes or functions, and shows the flow of data between them.

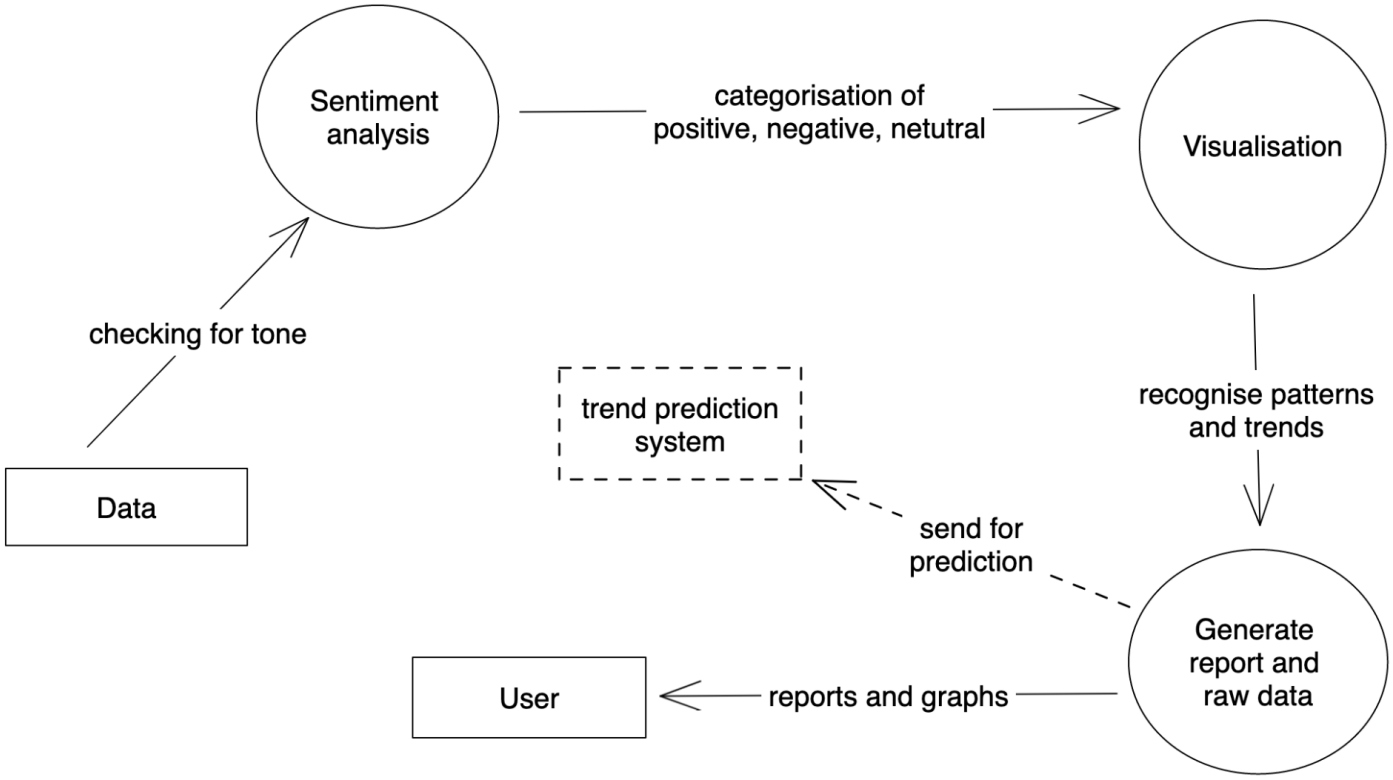


Level 2 (Detailed DFD):

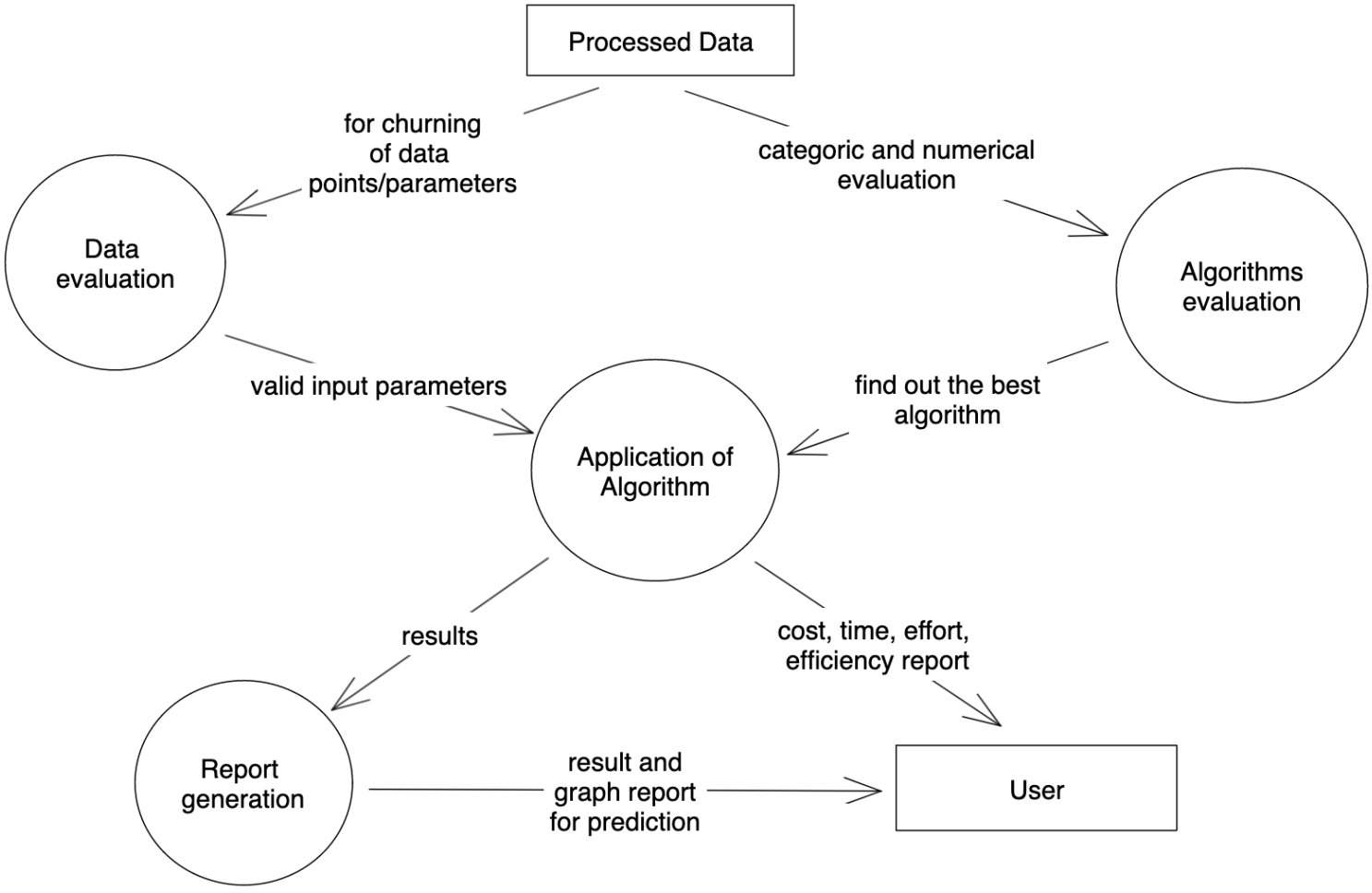
Level 2 - Detailed DFDs: Detailed DFDs are lower-level DFDs that provide more detail about specific processes or functions in the Top-Level DFD. They show the inputs, processes, and outputs of each process or function, and the data flows between them.



DFD level 2 for data collection



Level 2 DFD for sentiment analysis



Level 2 DFD for trend prediction

Data dictionary

A data dictionary is a document or a file that defines and describes the data elements and their relationships within a specific system. It provides a comprehensive list of data objects, their attributes, data types, and other relevant information such as allowable values, units of measure, and business rules. A data dictionary serves as a central reference point for data analysts, developers, and users to ensure consistency and accuracy in data usage across an organization.

Data Dictionnary:

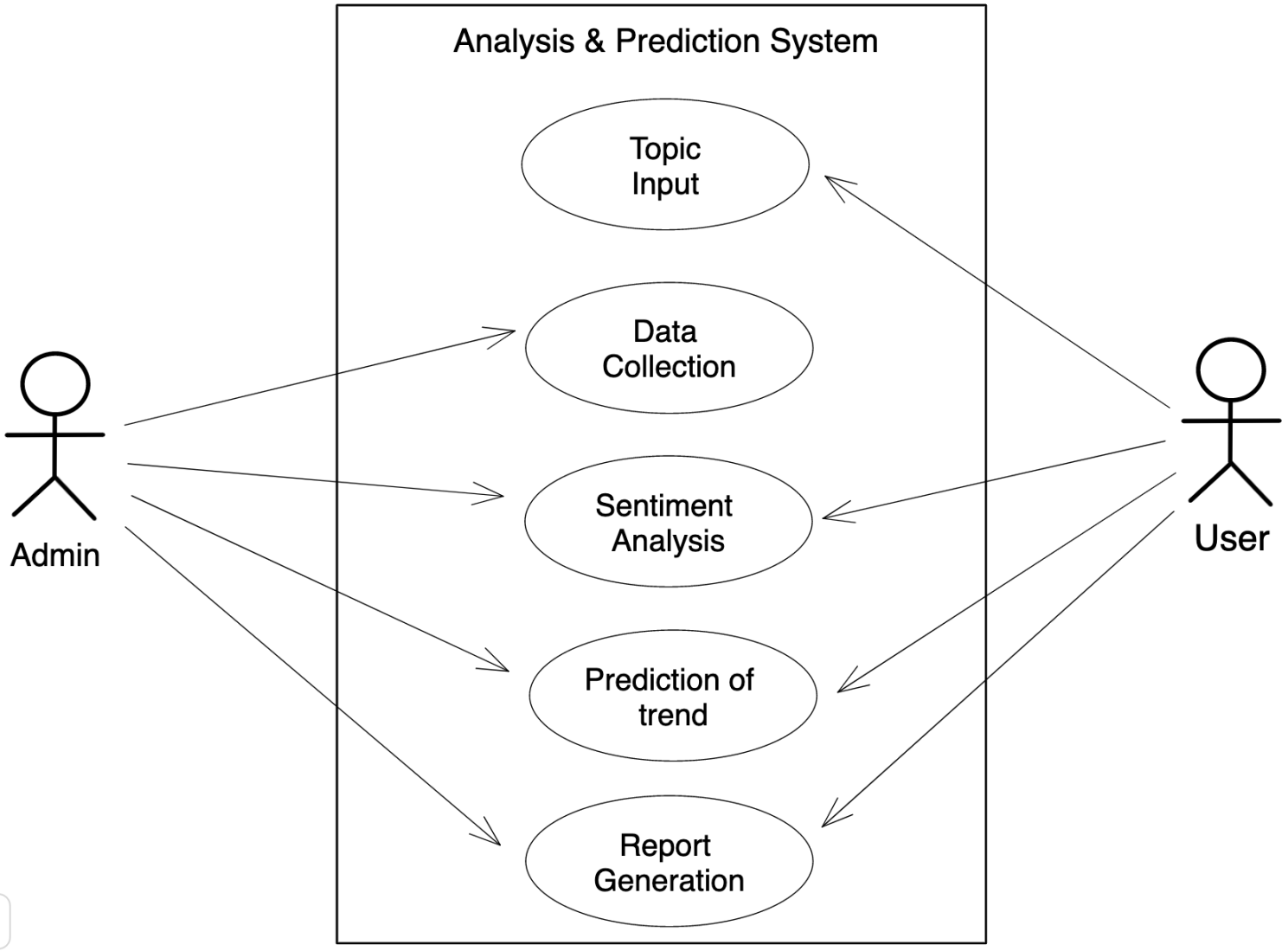
Field Name	Data Type	Data Format	Field Size	Description
Tweet	String	Alpha-numeric	300	The text content of a tweet.
Username	String	Alpha-numeric	10	The username of the Twitter user who posted the tweet.
Timestamp	String	Date/time	-	The date and time when the tweet was posted.
Emotion	String	Text	10	The emotion expressed in the tweet (positive, negative, or neutral).
Topic	String	Text	15	The topic that the tweet is related to.
Location	String	Text	50	The location of the Twitter user who posted the tweet.
Sentiment	String	Enum	10	The overall sentiment of the topic (positive, negative, or neutral).
Trend	Integer	Numeric	-	The predicted trend of the topic in the next few days or months.

This data dictionary defines the data elements used in the Twitter sentiment analysis project, including their data types and descriptions. It provides a clear understanding of the data and how it is used in the project, which will be useful for developers, analysts, and other stakeholders involved in the project.

By defining all of the data elements that are used in your project and specifying their data types and field sizes, you can ensure that your data is accurate and consistent. A well-defined data dictionary can also help you to communicate effectively with other members of your team and with stakeholders outside of your team.

Use Case Diagram

A use case diagram is a graphical representation of the interactions between actors (users or external systems) and a system. It shows the functionality of a system and how the different actors interact with it to achieve specific goals or tasks.
Use case diagrams are used to define and model the requirements of a system in a clear and concise way. They help to identify the different user roles and their interactions with the system, as well as the system's responses to those interactions. This makes it easier to understand and document the system's behavior and to communicate it to stakeholders, such as developers, testers, and users.



The use case diagram for this project outlines the main functionalities of the system and the actors or entities that interact with it. The primary use case is to analyze the sentiment trends of a specific topic on Twitter, which involves the use of the Twitter API and a Sentiment Analysis Model. The sentiment analysis model is used to classify tweets as positive, negative or neutral, which is then used to present the current sentiment trends related to the topic.

In addition to sentiment analysis, the system also has the capability to predict future sentiment trends and track brand reputation. The sentiment analysis model is also used to analyze customer feedback, which can be useful for businesses to improve their products or services. The system can also identify influencers based on the sentiment of their tweets and provide the results to the user.

Overall, the use case diagram highlights the different functionalities of the system and the various actors who can benefit from it, such as businesses, researchers, or individuals interested in a particular topic.

Use case description:

A use case description is a document that outlines the steps a user takes to achieve a specific goal with a system. It includes information such as preconditions, triggers, and post-conditions to define the scope of the use case and ensure that the system functions correctly. It helps to ensure that the system meets the user's needs and requirements.

1. **Topic Input**

1.1 Introduction

This use case lets the user enter the research topic into the system.

1.2 Actors

User

1.3 Pre-Condition

None

1.4 Post-Condition

If the use case is successful, the user input is registered into the system, else the state of the system does not change.

1.5 Flow of Events

1.5.1 Basic Flow

The system requests and validates the user input. If it is valid, the user's query is registered into the system and the user interface is displayed

2. **Data Collection**

2.1 Introduction

In this use case, based on the user's input, tweets are collected from twitter using a tweet scraping tool.

2.2 Actors

User, Admin

2.3 Pre-condition

User should've given some research topic

2.4. Post condition

If this use case is successful, tweets are collected and saved in memory in some way.

2.5. Flow of Events

2.5.1. Basic flow

User input is used for scraping tweets. If tweets are found then it is copied into a text file.

3. Sentiment Analysis

3.1 Introduction

Sentiment analysis is performed.

3.2 Actors

Administrator, User

3.3 Pre-condition

Dataset should be in correct format

3.4 Post-condition

None.

3.5 Flow of Events

3.5.1 Basic Flow

Sentiment of each tweet is rated through polarity and rated as positive, negative and neutral.

4. Prediction of trend

4.1 Introduction

This use case predicts the trend based on the analysis made in the previous step.

4.2 Actors

Administrator, User

4.3 Pre-condition

The analysis report should be made and categoric classification must be done.

4.4. Post condition

None.

4.5. Flow of Events

4.5.1. Basic flow

If the prediction is successfully performed, then the overall result are compiled and are presented to the user through next step.

5. Report generation

4.1 Introduction

This use case displays the overall sentiment of tweets, the trend prediction and patterns related to the research topic

4.2 Actors

Administrator, User

4.3 Pre-condition

Analysis and trend prediction should be performed with the desired topic.

4.4. Post condition

None.

4.5. Flow of Events

4.5.1. Basic flow

If everything performs as expected all the results should be generated.

Sequence Diagram

A sequence diagram is a type of interaction diagram that shows the interactions between objects or components in a system over time. It describes the flow of messages between objects and the order in which they occur to achieve a specific task or goal.

The diagram typically includes vertical lines, known as lifelines, that represent the objects or components involved in the interaction. The messages exchanged between the lifelines are represented as horizontal arrows with an attached label that specifies the type of message being exchanged. The sequence diagram can also include conditions, loops, and other constructs to represent complex interactions.

Sequence diagrams are a useful tool for modeling the dynamic behavior of a system and for identifying potential problems and bottlenecks in a system's design. They are widely used in software engineering to design and analyze complex systems, especially in the early stages of development when a system's architecture is being designed.

Abhi banaya nhi h
Thodi confusion h
baad m banata hu

Software requirement specifications

A software requirement specification (SRS) is a detailed document that outlines the functional and non-functional requirements of a software system. It serves as a foundation for the design and development of the software by specifying what the software should do and how it should behave under different circumstances. An SRS typically includes a description of the system's user interface, inputs, outputs, data requirements, processing functions, system interfaces, performance characteristics, and security requirements. The SRS also includes any constraints or limitations that must be considered during the development process. The document is typically created by a software development team, in collaboration with the client or stakeholders who will use or benefit from the software. It is an essential part of the software development process, as it helps ensure that the final product meets the needs and expectations of its users.

4.1 Overall description

The purpose of the software requirement specification (SRS) in this project is to provide a clear and detailed description of the functional and non-functional requirements for the development of the Twitter data analysis system. It outlines what the system should do and how it should behave, as well as any constraints or limitations that must be considered during the development process.

Overall, the SRS for this project will serve as a reference document for the development team and stakeholders, helping to ensure that the final product meets the needs and expectations of its users.

4.1.1 Product functions

Product function is the set of specific tasks that a software product must perform to meet the needs of its users. For example, if the software is a word processor, its product functions could include creating and editing documents, formatting text, and saving and printing files.

The product functions for this project can include:

1. Collecting Twitter data related to a specific topic.
2. Analyzing the sentiment of the tweets to determine public views (positive, negative, neutral) on the topic.
3. Generating trend analysis based on the sentiment analysis to predict the future trend for the topic.
4. Allowing the user to view the trend analysis and sentiment analysis on a user interface.
5. Providing the user with the ability to filter the data based on different criteria such as time period, sentiment, and keywords.

4.1.2 User Characteristics

User characteristics in SRS describe the characteristics of the end-users who will be using the software. This includes their technical knowledge, experience, and abilities, as well as any other relevant factors such as physical or cognitive limitations. It helps in designing software that meets the needs of the target audience.

The user characteristics for this project include:

- Basic knowledge of social media and Twitter usage
- Familiarity with data analytics and visualization tools
- Comfortable working with large datasets
- Familiarity with sentiment analysis and its application
- Knowledge of the topic being analyzed

4.1.3 General Constraints

General constraints refer to the limitations and restrictions that impact the development and implementation of a software system. These constraints may include technical limitations, legal requirements, or resource constraints.

Some general constraints regarding this project could include:

1. Data Privacy: The system should ensure that user data is handled securely and with the utmost confidentiality, adhering to privacy policies and regulations.

2. Scalability: The system should be designed in a way that it can scale up or down based on the user load and volume of data, without any loss of functionality or performance.

3. Maintainability: The system should be designed in a way that it can be easily maintained and upgraded over time, without causing any disruption to the user experience.

4. Reliability: The system should be reliable, with minimal downtime or system failures, to ensure that users can access it whenever they need to.

5. Legal compliance: The system should comply with all relevant legal requirements, such as data protection laws and intellectual property rights.

6. Time Constraints: The system should be delivered within a specified timeline and budget, as agreed upon by the stakeholders.

4.1.4 Assumptions and Dependencies

Assumptions and dependencies are those constraints that may not be very important to the system, however, any changes to this may affect the other requirements in the SRS.

The assumptions are described as follows:

- It is assumed that the user has a basic understanding of how to operate a computer and use the internet.
- It is assumed that the user has a stable internet connection to access the application.
- It is assumed that the user will provide accurate and valid information during the registration process.
- It is assumed that the user will comply with the terms and conditions of the application.

The dependencies are as follows:

- The application is dependent on the availability and performance of the database server.

- The application is dependent on the security and stability of the underlying operating system and web server software.

- The application is dependent on the compatibility of the user's device and web browser with the application's technology stack.

- The application is dependent on the availability and performance of any third-party APIs or services used in the application.

4.2 External Interface Requirements

External Interface Requirements are the requirements that specify how the system interacts with external entities or systems.

1. A user interface that allows users to input the topic of interest and view the sentiment analysis results and predictions. This could be a simple web or mobile application with basic user input and output fields.

2. Integration with the Twitter API or other data sources to automatically collect and preprocess the data. This could be done using a simple API key or authentication process.

3. Integration with existing sentiment analysis or machine learning libraries and tools. This could involve installing and configuring these tools on the server or cloud platform being used for the project.

4. Integration with existing visualization tools, such as matplotlib or D3.js, to create visualizations of the sentiment analysis results and predictions.

4.2.1 User Interface

UI is the front-end application view to which the client connects to utilise the software. The visual piece of a PC application or working framework through which a client cooperates with a PC or software. It decides how orders are given to the PC or the program and how information is shown on the screen. The user can log on to the collaborative page and access the results of the work performed.

The user interface (UI) for sentiment analysis project should be designed with the user in mind, keeping it simple and intuitive to use. A clean and uncluttered design with easy-to-understand labels and input fields can help users quickly understand what actions they need to take to get the desired results. Additionally, clear and informative visualizations can help users understand the sentiment analysis and prediction results, making it easier for them to draw meaningful insights from the data.

4.2.2 Hardware Interface

Hardware interface specifies the logical characteristics of each interface between the software product and the hardware elements of the system. This includes configuration characteristics such as number of ports, instruction sets, etc. It also covers such matters as what devices are to be supported, how they are to be supported, and protocols. For example, terminal support may specify full-screen support as opposed to line-by-line support.
The product needs to be connected to the internet for scraping tweets. At least 100 MB free hard disk space.

4.2.3 Software Interface

Specify the use of other required software products, and interfaces with other application systems.
User interface is part of software and is designed such a way that it is expected to provide the user insight of the software. UI provides a fundamental platform for human-computer interaction.For the project performed any software could work fine, python environment with all the required packages and libraries.
Software requirements for this sentiment analysis project would include a programming language, such as Python or Java, to implement the sentiment analysis and machine learning algorithms. Additionally, you may need to use external libraries and tools for data preprocessing, sentiment analysis, and visualization. Depending on the scale of the project, you may also need to use a cloud platform or server to handle the data processing and storage.

4.3 Functional Requirements

A Functional Requirement (FR) is a depiction of the help that the software should offer. It depicts a software framework or its part. A capacity is only contributions to the software framework, its way of behaving, and yields. It tends to be an estimation, information control, business process, client association, or whatever other explicit functionality which characterises which work a framework is probably going to perform. Functional Requirements in Software Engineering are additionally called Functional Specification.

4.3.1 FR1 Input Requirement

The user input is required on some research topic that needs to analysed and predicted using the system.

4.5 Design Constraints

Design constraints are limitations or restrictions that impact the design of a system or product. These constraints may be related to technical, practical, or regulatory considerations, and may include factors such as budget, time, resources, technology, safety, or compatibility with existing systems or standards.
Examples of design constraints in software engineering might include limitations on hardware capabilities, memory or processing power, available programming languages or development tools, or compliance with industry-specific regulations or standards.
By understanding and working within these constraints, designers and developers can create effective solutions that meet the needs of their users while still remaining feasible and practical to implement.

Some design constraints for the project might include:

1. Data availability: The availability and quality of Twitter data related to the chosen topic can be a significant constraint on the project. If there is limited or biased data available, the accuracy and effectiveness of the sentiment analysis and prediction models may be compromised.

2. Computing resources: The complexity of the sentiment analysis and machine learning algorithms used in the project can impact the computing resources required to run the analysis. Limited computing resources may constrain the project's ability to process large volumes of data or run more advanced analysis techniques.

3. Time constraints: The time available to complete the project may be limited, which can impact the scope and complexity of the sentiment analysis and prediction models. Limited time may require simplifying the analysis techniques or focusing on a smaller subset of the available data.

4. Regulatory compliance: There may be regulatory constraints related to privacy or data protection that need to be taken into account when collecting and analyzing Twitter data.

5. Technical expertise: The level of technical expertise required to implement the sentiment analysis and prediction models may be a constraint on the project. If the necessary skills are not available within the team, this may impact the choice of analysis techniques or require additional resources to be allocated for training or hiring.

Chapter 5 Estimations

Project estimation is the process of forecasting the effort, time, and resources required to complete a project successfully. It involves estimating the various project parameters such as cost, duration, scope, and quality, among others. Project estimation plays a crucial role in project planning and helps stakeholders make informed decisions about project budgeting, resource allocation, and scheduling.

5.1 Function points

Function Point (FP) is a component of programming improvement that assists with approximating the expense of advancement from the get-go all the while. It might gauge functionality according to the client's point of view.
The basic and primary purpose of the functional point analysis is to measure and provide the software application functional size to the client, customer, and the stakeholder on their request. The function point (FP) metric can be used effectively as a means for measuring the functionality delivered by a system.

Using historical data, the FP metric can then be used to:

1. Estimate the cost or effort required to design, code, and test the software.
2. Predict the number of errors that will be encountered during testing.
3. Forecast the number of components and/or the number of projected source lines in the implemented system.

The information domain values for this project have been mentioned in table below:

Information Domain Value	Count	Weighting Factor : Simple	Weighting Factor : Average	Weighting Factor : Complex
External Inputs [EIs]	1	3	4	6
External Outputs [EOs]	4	4	5	7
External Inquiries [EQs]	3	3	4	6
Internal Logical Files [ILFs]	3	7	10	15
External Interface Files [EIFs]	1	5	7	10

The definition of the terms used in above table are as follows :

- External Inputs (EIs): It is a transaction function in which the data goes into the application from outside the boundary to inside. This data is coming from external to the application.

- External Outputs (EOs): It is a transaction function in which data comes out of the system.

- External Inquiries (EQs): It is a transaction function with both input and output components which result in data retrieval.

- Internal Logical Files (ILFs): Internal Logical File (ILF) is a user identifiable group of logically related data or control information that resides entirely within the application boundary.

- External Interface Files (EIFs): It is a user identifiable group of logically related data or control information that is used by the application only for reference purposes.

To compute function points (FP), the following relationship is used:

$$FP = \text{count total} * [0.65 + 0.01 * \Sigma(Fi)]$$

where “count total” is the sum of all FP entries obtained from Figure 5.1

The Fi (i = 1 to 14) are value adjustment factors (VAF) based on responses to the 14 questions. Each of these questions is answered using an ordinal scale that ranges from 0 (not important or applicable) to 5 (absolutely essential). The constant values in the Equation/formula above and the weighting factors that are applied to information domain counts are determined empirically.

The value adjustment factors for this project have been mentioned in table below:

Q #	Question	Value
1	Does the system require reliable backup and recovery?	1
2	Are data communications required?	5
3	Are there distributed processing functions?	3
4	Is performance critical?	4
5	Will the system run in an existing, heavily utilised operational environment?	5
6	Does the system require online data entry?	5
7	Does the online data entry require the input transaction to be built over multiple screens or operations?	1
8	Are the master files updated online?	2
9	Are the inputs, outputs, files, or inquiries complex?	2
10.	Is the internal processing complex?	3
11	Is the code designed to be reusable?	2
12	Are conversion and installation included in the design?	3
13	Is the system designed for multiple installations in different organisations?	2
14	Is the application designed to facilitate change and ease of use by the user?	2

The VAF for our project are as follows:

$$\Sigma(Fi) = 1 + 5 + 3 + 4 + 5 + 5 + 1 + 2 + 2 + 3 + 2 + 3 + 2 + 2 = 40$$

Calculating the count total for the project:

Assumptions:

Number of external inputs (EIs) = 1
Number of external outputs (EOs) = 4
Number of external inquiries (EQs) = 3
Number of internal logical files (ILFs) = 3
Number of external interface files (EIFs) = 1
Weighting factor = simple

Count total = EIs * simple weighing factor (3)
+ EOs * simple weighing factor (4)
+ EQs * simple weighing factor (3)
+ ILFs * simple weighing factor (7)
+ EIFs * simple weighing factor (5)

= 3 + 16 + 9 + 21 + 5 = 54

Calculating FP:

FP = count total * [0.65 + 0.01 * Σ(Fi)]
= 54 * [0.65 + 0.01 * 40] = 54 * 1.05
= 56.7

5.2 Efforts

In software development, effort estimation is the process of predicting the most realistic amount of effort, in terms of person-hours or money, required to develop or maintain software based on incomplete, uncertain and noisy input.
It is calculated as E = FP / P where E stands for effort, FP stands for Function Points and P stands for productivity.

Here productivity, P = 17

Now,
Effort = FP/P

= 56.7 / 17
= 3.33 person-months

5.3 Cost

The cost estimate is the monetary spend that is done on the efforts to create and test software in Software Engineering. Cost assessment models are a few numerical calculations or parametric conditions that are utilised to estimate the cost of an item or an undertaking.

Here, Effort = 3.33 person-months
Burdened Rate Assuming = $500

Cost is calculated as Cost = Effort * Burdened Rate Assuming
= 3.33 * 500 = $1665

Chapter 6 Scheduling

A schedule in the project's time table consists of sequenced activities and milestones that are needed to be delivered under a given period of time.

TimeLine chart:

Timeline Chart required following phases of software creation:

1. Requirements gathering and analysis phase: This phase involves gathering and analyzing the requirements for the software, including functional and nonfunctional requirements, as well as user needs and business goals.

2. Design and architecture phase: This phase involves creating the software design and architecture, including defining the system components, data models, and interfaces.

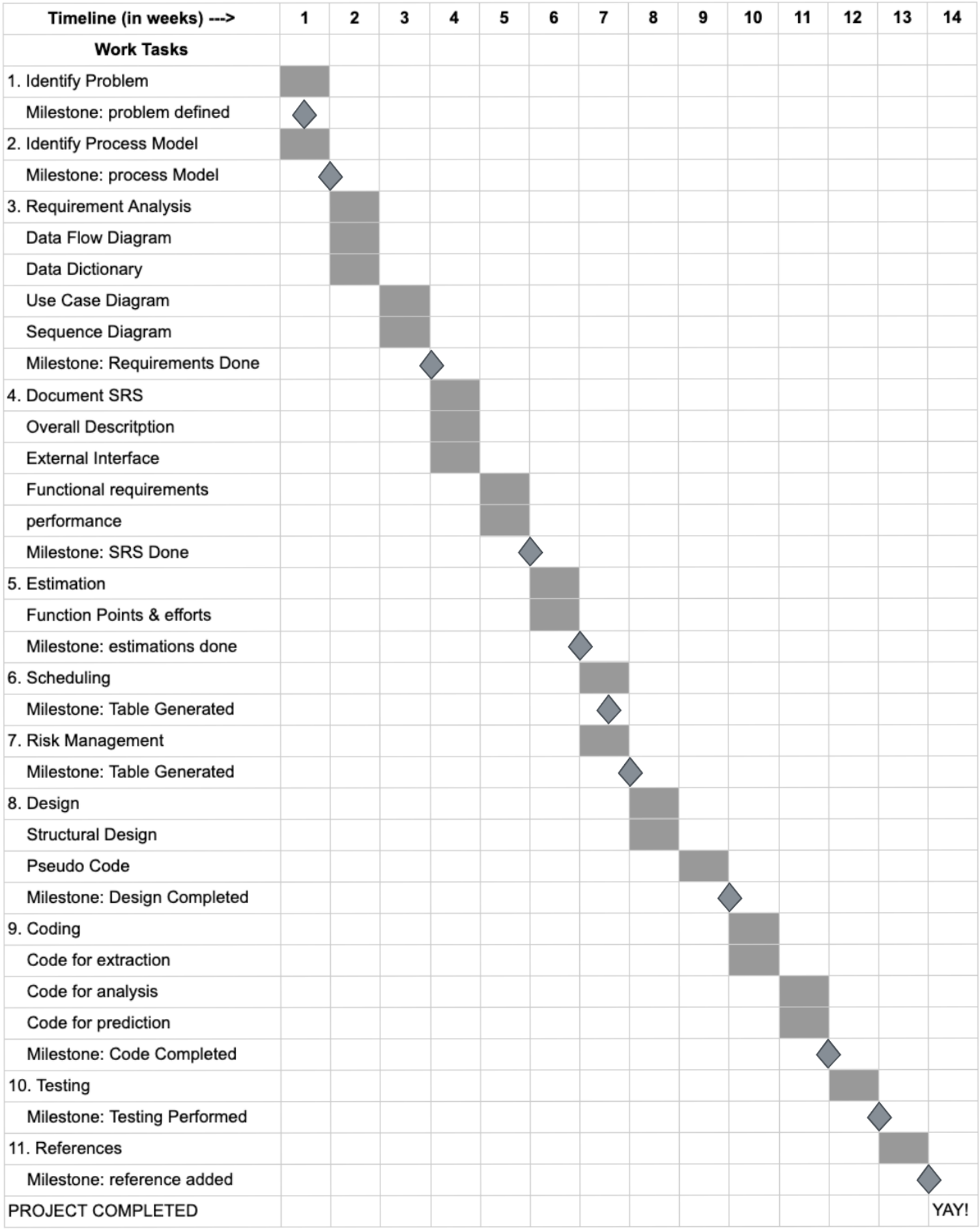
3. Development phase: This phase involves building the software components, integrating them into a working system, and performing unit testing.

4. Testing and quality assurance phase: This phase involves testing the software to identify defects and quality issues, as well as conducting code reviews, and implementing fixes.

5. Deployment and release phase: This phase involves deploying the software to production environments and releasing it to end-users.

6. Maintenance and support phase: This phase involves maintaining and updating the software, fixing bugs, and providing support to end-users.

The time line chart for this project is presented as follows:



Chapter 7 Risk Management

A software project may be subject to a wide range of risks. It is necessary to categorise risks in order to be able to systematically identify the significant risks that may affect a software project. The project manager can then determine which risks in each class are applicable to the project.

There are three main classifications of risks which can affect a software project:
Project risks: risks that threaten the plan of the project.
Technical risks: risks that threaten the quality and timeliness of the software to be produced.
Business risks: risks that threaten the viability of the software to be built.

Risk Table