SOCIAL MEDIA BLOG APP - APNABLOG

The Project submitted to
Sant Gadgebaba Amravati University, Amravati
towards partial fulfillment of the Degree of Bachelor
of Engineering
in
Information Technology

Guided by :

Prof. Ms. P.V.Kale

Submitted by :

Vivek Bhore
Rutik Gawande
Pratik Bondre
Vrushabh Guntiwar

DEPARTMENT OF INFORMATION TECHNOLOGY
SHRI SANT GAJANAN MAHARAJ COLLEGE OF ENGINEERING,
SHEGAON (M.S.)
2021- 2022

SHRI SANT GAJANAN MAHARAJ COLLEGE OF ENGINEERING
SHEGAON

2021-2022

CERTIFICATE

This is to certify that **Mr. Pratik Bondre, Mr. Vivek Bhore, Mr. Rutik Gawande, Mr. Vrushabh Guntiwar** students of final year B.E. (Information Technology) in the year 2021-2022 of Information Technology Department of this institute has completed the project work entitled "**SOCIAL MEDIA BLOG APP-APNABLOG**" based on syllabus and have submitted a satisfactory account of their work in this report which is recommended for the partial fulfillment of degree of Bachelor of Engineering in Information Technology.

Prof. Ms. P. V. Kale
Project Guide

Dr. A S Manekar
Head, Dept. of IT
S.S.G.M.College of Engineering,
Shegaon

Dr. S. B. Somani
Principal
S.S.G.M.College of Engineering,
Shegaon

SHRI SANT GAJANAN MAHARAJ COLLEGE OF ENGINEERING
SHEGAON

2021-2022

CERTIFICATE

This is to certify that the project work entitled "**SOCIAL MEDIA BLOG APP-APNABLOG**" submitted by **Mr. Pratik Bondre, Mr. Vivek Bhore, Mr. Rutik Gawande, Mr. Vrushabh Guntiwar** students of final year B.E. (Information Technology) in the year 2021-2022 of the Information Technology Department of this institute, is a satisfactory account of their work based on the syllabus which is approved for the award of Bachelor of Engineering in Information Technology.

Internal Examiner
Date:

External Examiner
Date:

ACKNOWLEDGEMENT

The real spirit of achieving goals is through the way of excellence and lustrous discipline. We would have never succeeded in completing our task without the cooperation, encouragement, and help provided to us by various personalities.

We would like to take this opportunity to express our heartfelt thanks to our guide ***Prof. Ms. P. V. Kale*** *for her esteemed guidance and encouragement, especially through difficult times. Her suggestions broaden our vision and guide us to succeed in this work. We are also very grateful for her guidance and comments while studyingpart of our project and learning many things under her leadership.*

We would also like to extend our sincere thanks to ***Prof. A. K. Shahade,*** *Project-In-Charge for his valuable support and feedback during the entire course of the project.*

We also extend our thanks to ***Dr. A. S. Manekar****, Head of Information Technology Department, Shri Sant Gajanan Maharaj College of Engineering, Shegaon for providing us with a variety of opportunities and inspirations to gather professional knowledge and material that made us consistent performers.*

We also extend our thanks to ***Dr. S. B. Somani****, Principal, Shri Sant Gajanan Maharaj College of Engineering, Shegaon for providing us the infrastructure and facilities without which it was impossible to complete this work.*

Also, we would like to thank all teaching and non-teaching staff of the department for their encouragement, cooperation, and help. Our greatest thanks toall those who wished us success, especially parents and friends.

1. ***Pratik Bondre***
2. ***Vivek Bhore***
3. ***Rutik Gawande***
4. ***Vrushabh Guntiwar***

ABSTRACT

‘ApnaBlog’ is going to be a social media blogging app in which any user can create and post his own written blog. The app will include different groups/categories such as sports, movies, technology and can be created more like these by the user. So, user will be able to post his blog in any of these category. When other users will visit website they can read the blog and either upvote or downvote it. The popularity of blog will depend on the votes it receives. In addition to this, using a machine learning technique a preview of every blog will be shown to reader so that he can choose weather to read it further or not. Also, a toxic and abusing comment detector also will be there capable of maintaining healthy discussion in comment section.The explosion of social media is changing the way communicate. In case of blogging platforms the current problems are like not having a clear topic and niche for the blog, not keeping on topic, absence of feedback or comments, inability to handle engagement, inability to handle offensive comments. So, through our social media app is are trying to deal with these kind of issues.Extractive summarization produces summaries by choosing a subset of the sentences in the original document(s). This contrasts with abstractive summarization, where the information in the text is rephrased. Although summaries produced by humans are typically not extractive, most of the summarization research today is on extractive summarization. Automating the process of identifying the abusive comments and blocking them would not only save time, but also provide safety to the users. Unfortunately, the social media platforms face these issues all the time and find it difficult to identify and stop these toxic remarks before it leads to the abrupt end of conversations. The problem is simple yet tiring to be accomplished through/read every comment to mark it as a Toxic or Abusive category this must be done systematically to get to the root of the problem. This can be filtered by checking the use of words like "bastard", "bitch", "shit" etc. can be flagged. Tenserflow (toxicity model) can be used for detection of toxic comment. As more and more people are expressing their opinions on the web, the Internet is becoming a popular and dynamic source of opinions. Natural language tools for automatically mining and organizing these opinions on various events will be very useful for blogging sites.

Keywords*: summarization, data mining, machine learning, toxic post, article, voice, recognition, speech, text.*

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1. INTRODUCTION

1.1 Preface

Social media is progressively exposed to issues of damaging conduct, for example, private assaults and cyberbullying. It is troublesome and time consuming to manually check which of the comments need to be blocked. Therefore, automating the process of identifying the abusive comments and blocking them would not only save time, but also provide safety to the users.

The elegance with online comment posting has grown at a faster rate and Social media is progressively exposed to issues of damaging conduct, for example, private assaults and cyber bullying. It is troublesome and time consuming to manually check which of the comments need to be blocked. Therefore, automating the process of identifying the abusive comments and blocking them would not only save time, but also provide safety to the users. Unfortunately, the social media platforms face these issues all the time and find it difficult to identify and stop these toxic remarks before it leads to the abrupt end of conversations. The problem is simple yet tiring to be accomplished through/read every comment to mark it as a Toxic or Abusive category this must be done systematically to get to the root of the problem. This problem can be addressed by training an ML model which deals with the comment section to flag a particular comment as abusive or not. The toxicity model detects whether text contains toxic content such as threatening language, insults, obscenities, identity-based hate, or sexually explicit language.

The model was trained on the civil comments dataset: https://figshare.com/articles/data_json/7376747 which contains 2 million comments labeled for toxicity. This can be filtered by checking the use of words like "bastard", "bitch", "shit" etc. can be flagged. Tenserflow (toxicity model) can be used for detection of toxic comment. As more and more people are expressing their opinions on the web, the Internet is becoming a popular and dynamic source of opinions. Natural language tools for automatically mining and organizing these opinions on various events will be very useful for blogging sites. Text summarization is the process of automatically creating a compressed version of a given text that provides useful information for the user. The information content of a summary depends on user’s needs. Topic-oriented summaries focus on a user’s topic of interest, and extract the information in the text that is related to the specified topic. On the other hand, generic summaries try to cover as much of the information content as possible, preserving the general topical organization of the original

text.In this project Extractive based text summarization is used for extracting summary from a blog. Extractive summarization produces summaries by choosing a subset of the sentences in the original document(s). This contrasts with abstractive summarization, where the information in the text is rephrased. Although summaries produced by humans are typically not extractive, most of the summarization research today is on extractive summarization. Purely extractive summaries often give better results compared to automatic abstractive summaries. This is due to the fact that the problems in abstractive summarization, such as semantic representation, inference and natural language generation, are relatively harder compared to a data-driven approach such as sentence extraction.

1.2 Statement of problem and objective

The explosion of social media is changing the way it communicate. In case of blogging platforms the current problems are like not having a clear topic and niche for the blog, not keeping on topic, absence of feedback or comments, inability to handle engagement, inability to handle offensive comments.So, through our social media app we is trying to deal with these kind of issues.

Extractive summarization produces summaries by choosing a subset of the sentences in the original document(s). This contrasts with abstractive summarization, where the information in the text is rephrased. Although summaries produced by humans are typically not extractive, most of the summarization research today is on extractive summarization.

1.3 Scope and Limitations of the Project

There are some limitations in business model which are as follows:

1. It gives list of all the project goals, tasks, deliverables and deadlines as a part ofthe planning process.
2. Details all the boundaries of the project while also establishing the responsibilitiesof the team,
3. Provides “Done”, “In-Progress”, and “On Hold” states for clarity on project status.
4. Any updates to the data by team members or the Project coordinator or guidecannot see immediately by the rest of the team.

1.4 Organization of the Project

The project is organized as follows:-

1. Chapter 1 gives Introduction to the project
2. Chapter 2 gives Literature survey of the project.
3. Chapter 3 provides analysis of project.
4. Chapter 4 provides design phase of project.
5. Chapter 5 provides how project is implemented
6. Chapter 6 gives conclusion.
7. Chapter 7 discuss about Future work.

CHAPTER 2

LITERATURE SURVEY

2. LITERATURE SURVEY

The main aim of this project is to create a social media web progressive web application. **Progressive Web Apps** (PWAs) are web apps that use service workers, manifests, and other web-platform features in combination with progressive enhancement to give users an experience on par with native apps.. PWAs are built using front-end technologies such as HTML, CSS and JavaScript and bring native-like user experience to the web platform. PWAs can also be installed on devices just like native apps.

Classification of Online Toxic Comments Using Machine Learning Algorithms [1]

A huge amount of data is released daily through social media sites. This huge amount of data is affecting the quality of human life significantly, but unfortunately due to the presence of toxicity that is there on the internet, it is negatively affecting the lives of humans. Due to this negativity, there is a lack of healthy discussion on social media sites since toxic comments are restricting people to express themselves and to have dissenting opinions. So, it is the need of the hour to detect and restrict the antisocial behavior over the online discussion forums. Although, there were efforts in the past to increase the online safety by site moderation through crowd-sourcing. schemes and comment denouncing, in most cases these techniques fail to detect the toxicity. So, to find a potential technique that can detect the online toxicity of user content effectively. As Computer works on binary data and in real-world data in various other forms i.e. images or text. Therefore, to convert the data of the real world into binary form for proper processing through the computer.

Text classification can be easily applied on given data set and set of labels by applying the data on a function, that will assign a value to each data value of data set. In this context, another scientist research introduced a technique that incorporates crowdsourcing and machine learning to evaluate on-scale personal attacks. Recently, a project called perspective was introduced by Google and Jigsaw, to detect the online toxicity, threats, and offensive content with the help of machine learning algorithms. In another approach, Convolutional Neural Networks (CNN) was used in text classification over online content , without any knowledge of syntactic or semantic language. In the approach used by Y. Chen et al., introduced a combination of a parser and lexical feature to detect the toxic language in YouTube comments to protect adolescents. In the approach used by Sulke et al., Online comments are classified with the help of machine learning algorithms. So, lots of work has already been done to detect and classify online toxic comments. In our research paper,will

learn from the already published work and use machine learning algorithms to detect and classify online toxic comments with better accuracy.

In another study five transformations used namely URLs features reservation, negation transformation, repeated letters normalization, stemming and lemmatization on twitter data and applied linear classifier available in WEKA machine learning tool. They found the accuracy of the classification increases when URLs features reservation, negation transformation and repeated letters normalization are employed while decreases when stemming and lemmatization are applied. Other studies investigated the effect of transformation on five different twitter datasets in order to perform sentiment classification and found that removal of URLs, the removal of stop words and the removal of numbers have minimal effect on accuracy whereas replacing negation and expanding acronyms can improve the accuracy. Most of the exploration regarding application of the transformation has been around the sentiment classification on twitter data which is length-restricted. The length of online comments varies and may range from a couple of words to a few paragraphs.

In another study, it shows the impact of transformation on text classification by taking into account four transformations and their all possible combination on news and email domain to observe the classification accuracy. Their experimental analyses shown that choosing appropriate combination may result in significant improvement on classification accuracy. Another study used normalization of numbers, replacing very long unknown words and repeated punctuations with the same token, explained the role of transformation in sentiment analyses and demonstrated with the help of SVM on movie review database that the accuracies improve significantly with the appropriate transformation and feature selection. They used transformation methods such as white space removal, expanding abbreviation, stemming, stop words removal and negation handling. Other works focus more on modeling as compared to transformation.

Extractive Text Summarization using sentence ranking [2], In earlier researches, Summarization was done on scientific documents based on the proposed features like phrase frequency, word (Luhn, 1958) , key phrases (Edmundson, 1969) and position in the text (Baxendale, 1958) [22] .

In 1958, most of the earlier works are done on the single document mainly focusing on technical document. he has done his research on the extractive summarization. In his research he extracted important sentence by calculating word frequency and phrase frequency that gives the useful measure of its significance.

The extraction of important sentence by using the position of text. The author has tested 200 paragraphs towards his goal to find that in 85% of the sentences which author has taken first topic which is main topic sentence and the last sentence came 7%. The most accurate sentence would be selected from these two sentences.

Another paper involved research on extracted summarization in this he extracted important sentence by using two features position and word frequency importance were taken from the previous works. The author has added two they are: presence of cue words, and the skeleton of the document.

A. Graph based methods

In these methods, every sentence of a document is represented as a node of the graph and the relation between the sentences are denoted as edge. Every node is scored based. on the structured and non-structured text features, and the similarity between the sentences helps in traversing the graph in a significant manner. The extraction based multi-document summarization (Sripada et al., 2005) employs the WordNet based semantic similarity to find the similarity scores and to compute the sentence importance using ten text features. The unified approach for multi-document summarization relies on four assumptions to find the locally and globally important sentences in the documents through the affinity matrix of the sentence similarity (Wan, 2010) [17]. The G-FLOW (Christensen et al., 2013) [15] is a system for coherent extractive multidocument summarization that generates an ordered summary by optimizing the coherence and salient factors, where the coherence of text is evaluated by an approximated discourse graph. Glavas and ˇ Snajder (2014) [14] introduce an event based ˇ summarization method that exploits the strength of machine learning rule based approaches and performs effectively on the event oriented document collection. The complex networks are also based on the graph theory. Amancio (2015) [13] explores the linguistic properties for short written texts, which may be very helpful in summarization task. The topological properties of complex networks in short texts are investigated and it has been found that these can improve the global characterization of long texts also. Amancio et al. (2012)[13] discuss a summarization method based on complex networks and syntactic dependencies. They employ some new metrics such as betweenness, vulnerability, closeness, and diversity, to extract the sentences from documents and they find that the

diversity metric is the best for extraction. It is further suggested that the syntactic parsing can enhance the performance of summarizers. Some research discuss a multilayer approach based extractive summarizer where several measurements such as degree, strength, page rank, accessibility, symmetry, shortest path, absorption time, etc., are used to weight the edges in the network of documents. They find that the distinction between intra- and inter-layer edges can play a major role in improving the results of a summarizer. Few researches present a survey on the graph based summarization methods by categorizing the state-ofthe-art methods according to the input graphs and associated approaches. This survey leads to several open research areas such as temporal graph summarization for document, improvements in standardizing, generalizing algorithms, etc. The general limitation with these methods is that these are poorly applicable to large scale of data. So, their performance may be limited to single document summarization. To address this limitation, propose the summarization methods based on meta-heuristic approaches, which can be efficiently applied to the large size of documents.

B. Maximal Marginal Relevance based methods

In maximum marginal relevance (MMR) based methods, the summarization task is modeled in such a way that the contents of produced summary should consist of relevant information to the query as well as minimal similarity among the contents. Goldstein and Carbonell (1998) [21] discuss a maximal marginal relevance based method for multi-document text summarization. This method balances the coverage and relevancy with query factors in the summary. The balancing weight for relevancy is taken as 0.7, and 0.3 for nonredundancy. Goldstein et al. (2000) [20] focus on the issues in multi-document summarization that are compression, speed, redundancy, and passage selection. Wang et al. (2009) discuss a summarization method for e-mail data by analyzing the relationship between the MMR model and content cohesion in e-mails that enhances the precision scores, present an MMR model with Naive-based tone biasing model. The content generated from MMR model is used as an input for the Naive biasing model using a set of polarity tags. Some of the MMR based methods are discussed in meta-heuristic based methods that consider them as an optimization problem. The general limitation with these methods is that their performances do not guarantee for the presence of both coverage and non-redundancy aspects in the summary. To address this limitation, the propose is clusterring based summarization methods that address both aspects.

C. Meta-heuristic based methods

For last couple of years, many researchers have focused on the optimization algorithms such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Harmony Search (HS), Differential Evolution (DE), and Cat Swarm Optimization (CSO) for single as well as multi-document summarization. To our best knowledge, the genetic algorithm was used first time in the document summarization task in (He et al., 2006) to retrieve the relevant sentences based on four summary factors: satisfied length, high coverage, high informativeness, and low redundancy. This method takes into account the similarity between the words using the WordNet that is further used to find the term frequency. Kogilavani and Balasubramanie (2010) [] present a method for optimal summary generation by grouping the related documents into a cluster and extracts the important sentences from each cluster using the genetic algorithm. The multi-criteria optimization based multidocument summarization finds the extractive generic summary with maximal relevance and minimal redundancy. The sentences are scored using five features: TFIDF, aggregate cross sentence similarity, title similarity, proper noun, and sentence length. Few studies discusses a technique for summarization using the cuckoo search using three features: coverage, cohesion, and readability for summarization. Alguliev et al. (2013) [16] discuss a summarization technique based on differential evolution that focuses on three aspects of summarization: content coverage, diversity, and length of summary; and optimizes these aspects using differential evolution. The cosine function is used for similarity measurement. A PSO based text summarization model focuses on maximizing the content coverage and minimizes the redundancy in the summary. This method optimizes the relevancy, redundancy, and summary length together through a binary PSO and the Google hit based dissimilarity function (NGD). The limitation of these methods is that the metaheuristic approaches used for the summarization generally get trapped into local optima. Moreover, these techniques do not provide information about the behavior of a function such as steepness and extrema in the search space. Thus, a gradient based optimization approach is used in the proposed method so that the convergence of the search algorithm can be drastically enhanced.

D. Other methods

In recent years, some summarization methods have been discussed based on Reinforcement Learning (RL). The main objective of RL is to facilitate the optimization of nondifferentiable functions such as ROUGE. In this respect, Narayan et al. (2018) [] discuss

an extractive summarization which is globally trained by optimizing the ROUGE function. It consists of three main modules: sentence encoder (implemented with Convolutional Neural Networks (CNNs) for continuous representation of sentences), document encoder (implemented with Recurrent Neural Networks (RNNs) + Long Short-Term Memory (LSTM) to avoid the vanishing gradient problem), and sentence extractor (implemented with RNNs + LSTM for summarization). It also uses the combination of maximum-likelihood cross-entropy loss and rewards obtained by the policy gradient RL as the objective function, discusses an RL+ML based model for abstractive summarization where a key attention mechanism and learning objective to address the redundancy problem has been introduced. Here, the sentence encoder and decoder are based on RNNs. Experimentally, this model is effective for long document summarization that suggests ROUGE should not be the only metric to optimize in summarization. Another study discuss a deep Q-Network based single document summarization model that uses both content and position based embedding features to select the sentences for summary. Here, it is presented some state-of-the-art summarization methods that are based fuzzy, evolutionary, and clustering algorithms and their hybrid form, presents a model for abstract generation that uses four processes: symbolic morphological, syntactic, semantic, and pragmatic processes. Each process is responsible for collecting the specific feature based information that is passed to an artificial neural network (ANN) to score the textual units. A model that exploits the strengths of summarist and PULSUMM base tree-based summarization systems, a method that combines K-mixture term weighting method and linguistic method to generate the summary. A summarization model combines the cohesive properties of the text with coherence relations and strengthens the lexical chain based summarizer using the rhetorical and argumentative structures, a hybrid summarization model for Spanish text that integrates the Cortex (inspired by linguistic technique) as well as Enertex (inspired by statistical physics) for summarization. Another study [8] discuss two hybrid models by exploiting the strength of fuzzy logic, evolutionary algorithm, and maximal marginal importance algorithm, and evaluate them on the DUC02 dataset. Their is a summarization model based on fuzzy logic, evolutionary

algorithms, and cellular learning automata (CLA) in which the similarity function is modeled as a combination of CLA and artificial bee colony (ABC) problem. The weights of text features are customized by using the PSO and GA and it is compared with fourteen other summarization methods. Mehta and Majumder (2018) [4] present a comparative study of various existing text summarizers with respect to the sentence ranking, sentence similarity, and text representation. They suggest that the combination of different techniques

(or a hybrid model) of summarization in a systematic manner can enhance the performance of the summarization system. Goularte et al. (2019) [3] discuss a method for text summarization using the fuzzy rules which is further used for automatic text assessment. They show that the fuzzy based summarization system can successfully enhance the quality of the generated summaries, a clustering based summarization method for opinion data (Opinosis) where they suggest that the clustering of reviews can play a major role in coverage of reviews related to every instance. Few researches discusses on nine heuristic based methods for sentence extraction from long documents. They suggest that the removal of redundant contents from a document can speed up the ability of summarization system for finding relevant sentences and summary and on a summarization method based on soft computing techniques that cluster the sentences of a document for finding similar sentences and merge them according to their similarities, A multi-document summarization method based on group sparse learning which acquires structural knowledge among the group of sentences. They use the Nesterov's method to optimize the group sparse convex for better convergence behavior, summarization method, called Heterogeneous Feature Symmetric Summarization (HFSS) is also one approach. A query based summarization method, called QMOS, was presented which is a two-stage procedure: sentiment analysis and summarization. The semantic sentiment analysis is carried out by combining multiple sentiment lexicons. The summarization is done on the basis of syntactic and semantic analysis of sentences. Azmi and abstractive summarization approach for Arabic text document, which uses a sentence reduction approach and rhetorical structural theory based sentence extraction approach to generate summary. A survey on swarm intelligence (SI) is presented which is based summarization techniques and report that the usage of SI approaches is quite limited with respect to summarization task. They discuss a summarization framework to cover multi-objective optimization task using SI. Sanchez-Gomez et al. (2018) [5] discuss a multi-objective ABC based summarization method and its efficacy is shown on the DUC02 dataset.

In this paper author has reviewed different techniques of Sentiment analysis and different techniques of text summarization. Sentiment analysis is a machine learning approach in which machine learns and analyze the sentiments, emotions present in the text. The machine learning methods like Naive Bayes Classifier and Support Machine Vectors (SVM) are used. these methods are used to determine the emotions and sentiments in the text data like reviews about movies or products. In Text summarization, uses the natural language processing (NPL) and linguistic features of sentencesare used for checking the importance

of the words and sentences that can be included in the final summary. In this paper, a survey has been done of previous research work related to text summarization and Sentiment analysis, so that new research area can be explored by considering the merits and demerits of the current techniques and strategies. Harsha Dave et al. [12] In this paper author has proposed a system to generate the abstractive summary from the extractive summary using WordNet ontology. The multiple documents had been used like text, pdf, word files etc. The author has discussed various text summarization techniques then author discussed step by step the multiple document text summarization approaches. The experiment result is compared with the existing online extractive tools as well as with human generated summaries and shows the proposed system gives good results. At last the author proposed for the future that the

In some researches the author introduces graph-based method LexRank. In this, the sentence score is calculated based on Eigenvector Centrality. It is cosine transform weighting method. In this, the original text is split into sentences and a graph is built where sentences act as the nodes. The complete method is explained in the paper. The results show that LexRank outperforms the existing centroidbased methods. This method is also performed well in case of noisy data. This method generates an extractive summary of the text.

Kavita Ganesan et al. [18] In this research paper the author proposed graph-based text summarization framework Opinosis. It generates abstractive summaries. Opinosis works on redundant data like human reviews on movies or products and provides abstractive summaries. Firstly, it creates the direct Opinosis-Graph of the text. Where nodes represent the word units of the text. Three unique graph properties: Redundancy capture, Collapsible structures and Gapped subsequence capture is used to explore and explore different sub-paths that help in the creation of abstractive summaries of the text. The valid path is selected and marked with high redundancy score, collapsed path and summary generation. Then all paths ranked in descending order according to scores. The duplicate paths are removed using Jaccard measure the results are compared with human summaries. Results show Opinosis summaries has better agreement with human summaries. For future work author proposed to use a similar idea to overlay parse trees, the author uses the extractive text summarization. The author gives the Wikipedia Articles as input to the system and identifies text scoring. Firstly, the sentences are Tokenized through pattern matching using regular expressions. Then getting data in form of set of words then stop words are removed from the set of words. The words are then stemmed. Then traditional methods are used for scoring of the sentences.

Scoring helps in classifying the sentences if they included in summary or not. It is found that scoring sentences based on citation give better results.

N. Moratanch et al. [11] In this paper the author presents an exhaustive survey on abstraction based text summarization techniques. The paper presents a survey on two broad abstractive summary approaches: Structured based abstractive summarization and Semantic-based abstractive summarization. The author presents the review of various researches on both approaches of abstractive summarization. The author also covered the various methodologies and challenges, in abstractive summarization.

In some paper the author proposed a particular version of KNN (K Nearest Neighbor) where the words are assumed as features of numerical vectors represents text. The similarity between feature vectors is computed by considering the similarity among attributes as well as among values. Text summarization viewed as the task of classification. The text is partitioned into paragraphs or sentences. Each paragraph or sentence is classified into 'summary or 'nonsummary' by the classifier. The sentences which are classified into 'summary' are extracted as results from summarizing the text and other text rejected. Improved results are obtained with the proposed version of KNN in text classification and clustering. The modified version of KNN leads to a more compact representation of data item and better performance.

Omar Sharif Iand Mohammed Moshiul Hoque: "Identification and Classification of Textual aggression in Social Media: Resource creation and evaluation" In this paper, the aggressiveness of the Bengali text is finished into various categories like religious, gendered, verbal, and political aggression class. Several classification algorithms such as LR, RF, SVM, CNN, and BiLSTM are implemented and also the developed model gained maximum accuracy when CNN and BiLSTM were combined.

"A Study of fastText Word Embedding Effects in Document Classification in Bangla Language" In this paper, the Fast text embedding technique was brought into the limelight for general text classification and was observed that with fastText word embedding significant performance are often gained without some preprocessing like lemmatization, stemming, and others. From the above papers, it may be concluded that between CNN and RNN approach CNN works best and faster for text sentiment classification and character level embedding while requires a lots of training but still gives the most effective results as

compared to word level or sentence level embedding. Some research papers experimented with different word embedding methodology in detailed comparison between the GLOVE model, Word2Vec, Doc2vec, and TF-IDF model. Out of which word2vec and glove model have shown effective results, but didn't label the typing mistakes and slangs. But with the newest technique named Fast text word embedding works better even with slang and typos, "Identification and Classification of Textual aggression in Social Media: Resource creation and evaluation" In this paper, the aggressiveness of the Bengali text is finished into various categories like religious, gendered, verbal, and political aggression class. Several classification algorithms such as LR, RF, SVM, CNN, and BiLSTM are implemented and also the developed model gained maximum accuracy when CNN and BiLSTM were combined.

"A machine learning approach to comment toxicity approach":This paper introduces the Machine Learning approach with Natural Language Processing to work out the various kinds of toxicity using 6 headed Machine Learning TF-IDF model. It had been observed that the model gives a mean validation accuracy of about 98.08% and absolute validation accuracy of 91.61%, "Detecting and Classifying Toxic Comment" During this paper, Support Vector Machine(SVM), Long-Short Term Networks(LSTM), Convolutional Neural Network(CNN) and, Multilayer Perceptron(MLP) methods are done at the word and character level embedding for identifying the toxicity and was observed that on word-level classifications best results were obtained with LSTM model and for character level classification CNN model works the best.

"Classification of online toxic comments using the logistic regression and neural networks models": In this paper 4 models were proposed: Logistic Regression model and three neural network models: Convolutional neural network (CNN), Long Short term memory (LSTM) with RNN, and Conv + LSTM and was concluded that the convolutional model works better than RNN models and also the best results were obtained when both Conv+ LSTM (2 LSTM layers and 4 Convolutional Layers) model worked together.

"A Comparative Study": The paper employed 3 different feature engineering techniques (Bigram with TFIDF, word2vec, and doc2vec) and eight machine learning algorithms (Naïve Bayes, Support Vector Machine, K Nearest Neighbor, Decision Tree,

Random Forest, Adaptive Boosting, Multilayer Perceptron, and Logistic Regression) to classify comments and was noticed that Bigram with TFIDF maintains the sequence of

words and then outperforms Word2vec and Doc2vec. Moreover, Support Vector Machine and Random Forest algorithms showed better results compared to the rest with KNN giving the least accuracy.

CHAPTER 3
ANALYSIS

3. ANALYSIS

3.1 Detailed Statement of the problem

Since March 2020, the world has been locked indoors due to the ongoing Covid-19 pandemic. Amidst this Covid-19 lockdown, many of the people have switched to the increasing use of social media websites and other apps to pass on their spare time. Now living in a social media driven world, there is no escaping it, so ignoring social media's potential to advance and enhance the blog is a dangerous oversight. In case of blogging platforms the current problems are like not having a clear topic and niche for the blog, not keeping on topic, Finding the right topic for the blog website is the very first and probably most important step in blogging. A lot of beginners fail here because of the unwillingness to narrow down their blog's focus. absence of feedback or comments, Most negative comments left on Social Media are complaints left by unhappy users inability to handle engagement, inability to handle offensive comments, Offensive comments should be removed and offenders should be warned or even blocked if necessary.Enforcing rules will not only help keeping the page cleaner, offenders will think twice before posting again, but it will also help other users feel safer especially in situations where the target is another user.

3.2 Requirement Specifications

A software requirements specification (SRS) is a description of a software system to be developed. The software requirements specification lays out functional and nonfunctional requirements, and it may include a set of use cases that describe user interactions that the software must provide to the user for perfect interaction. Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules. To derive the requirements, the developer needs to have a clear and thorough understanding of the products under development. This is achieved through detailed and continuous communications with the project team and customer throughout the software development process.

Hardware Requirements (minimum)

- System: Intel core 2 duo
- OS: Windows XP
- RAM: 1 GB
- Hard Disk: 20GB

Software

- Windows OS
- Visual Studio IDE
- Web Browser
- Computer System
- Node LTS
- MongoDB Atlas
- Internet
- Personal information of students and faculties from college.

3.3 Functional Requirements

A functional requirement defines a function of a system or its component, where a function is described as a specification of behavior between outputs and inputs. Basically, functional requirements are the statements that indicate what a system needs to do in order to provide a capability.

1. User can create blogs.
2. User can update or delete his/her blog.
3. User can search blogs according to various categories and through voice search.
4. User can read all blogs.
5. User can add comment and edit his/her comment.
6. Admin can edit/delete/add comment
7. Admin can delete a blog.

3.4 Non Functional Requirement

The non functional requirements elaborate a performance characteristic of the Non functional requirements of this project are :

1. Accessibility: project management and progress tracking system will beeasily accessible by Hod, faculty, project coordinator, and team members.

2. Availability: The web application will be available 24*7 and it can beaccessed anytime.

3. Usability: The project management and progress tracking system has an

4. Interactive User Interface.

5. Efficiency: The project management and progress tracking system is a veryefficient web application.

Non-Functional Requirements are the constraints or the requirements imposed on the system. They specify the quality attribute of the software. Non-Functional Requirements deal with issues like scalability, maintainability, performance, portability, security, reliability, and many more. Non-Functional Requirements address vital issues of quality for software systems. If NFRs not addressed properly, the results can include:

- Users, clients, and developers are unsatisfied.
- Inconsistent software.
- Time and cost overrun to fix the software which was prepared without keeping NFRs in mind.

Advantages of Non-Functional Requirement :

- They ensure the software system follows legal and adherence rules.
- They specify the quality attribute of the software.
- They ensure the reliability, availability, performance, and scalability of the software system
- They help in constructing the security policy of the software system.
- They ensure good user experience, ease of operating the software, and minimize.

3.5 Feasibility Study

A feasibility study is carried out to select the best system that meets the performance requirements. Feasibility is the determination of whether or not a project is worth doing. The process followed in making this determination is called a feasibility study. This type of study determines if a project can and should be taken. Since the feasibility study may lead to the commitment of large resources, it becomes necessary that it should be conducted competently and that no fundamental errors of judgment are made. Depending on the results of the initial investigation, the survey is expanded to a more detailed feasibility study. Feasibility study is a test of a system proposal according to its work-ability, impact on the organization, ability to meet user needs, and effective use ofresources. The objective of the feasibility study is not to solve the problem but to acquire a sense of its scope . During the study, the problem definition is crystallized and aspects of the problem to be included in the system are determined.

Consequently, costs and benefits are described with greater accuracy at thisstage. It consists of the following:

- Statement of the problem: A carefully worded statement of the problem that led to analysis.
- Summary of finding and recommendations: A list of the major findings and recommendations of the study. It is ideal for the user who requires quick access to the results of the analysis of the system under study. Conclusions are stated , followed by a list of the recommendations and a justification for them .
- Details of findings : An outline of the methods and procedures under-taken by the existing system, followed by coverage of the objectives and procedures of the candidate system. Included are alsodiscussions of output reports, file structures, and costs and benefits of the candidate system .
- Recommendations and conclusions: Specific recommendations regarding the candidate system, including personnel assignments, costs, project schedules, and target dates.

a) Technical Feasibility

Technical feasibility includes whether the technology is available in the market for development and its availability. The assessment of technical feasibility must be based on an outline design of system requirements in terms of input, output, files, programs and procedures. This can be qualified in terms of volumes of data, trends, frequency of updating, cycles of activity etc., in order to give an introductionof technical system.

b) Economic Feasibility

The economic feasibility study represents tangible and intangible benefits from the project by comparing the development and operational cost. The technique of cost benefit analysis is often used as a basis for assessing economic feasibility. This system needs some more initial investment than the existing system, but it can be justifiable that it will improve quality of service. Thus, feasibility study should centre

along the following points:

- Improvement resulting over the existing method in terms of accuracy,timeliness.
- Cost comparison.
- Estimate on the life expectancy of the hardware.

c) Behavioural/ Operational Feasibility

This analysis involves how it will work when it is installed and the assessment of political and managerial environment in which it is implemented. People are inherently resistant to change and computers have been known to facilitate change. The new proposed system is very much useful to the users and therefore it will accept broad audience from around the world.

3.6 Use Case Diagrams

To model a system the most important aspect is to capture the dynamicbehaviour. To clarify a bit in details, dynamic behaviour means the behaviour of the system when it is running or operating. So only static behaviour is not sufficient to model a system rather dynamic behaviour is more important than static behaviour. In UML there are five diagrams available to model dynamic nature and use case diagramis one of them. Now as it has to

discuss that the use case diagram is dynamic in nature there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors. So use case diagrams are consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system. So to model the entire system numbers of use case diagrams are used.

In the Unified Modelling Language (UML), a use case diagram can summarize the details of system's users (also known as actors) and their interactions with the system. To build one, use a set of specialized symbols and connectors. An effective use case diagram can help team discuss and represent:

- Scenarios in which system or application interacts with people, organizations, or external systems
- Goals that system or application helps those entities (known as actors) achieve
- The scope of the system

A use case diagram doesn't go into a lot of detail—for example, don't expect it to model the order in which steps are performed. Instead, a proper use case diagram depicts a high-level overview of the relationship between use cases, actors, and systems. Experts recommend that use case diagrams be used to supplement a more descriptive textual use case.

UML is the modeling toolkit that can use to build diagrams. Use cases are represented with a labeled oval shape. Stick figures represent actors in the process, and the actor's participation in the system is modeled with a line between the actor and use case. To depict the system boundary, draw a box around the use case itself.

UML use case diagrams are ideal for:

- Representing the goals of system-user interactions
- Defining and organizing functional requirements in a system
- Specifying the context and requirements of a system
- Modeling the basic flow of events in a use case

- Actors: The users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with application or system. They must be external objects that produce or consume data.

- System: A specific sequence of actions and interactions between actors and the system. A system may also be referred to as a scenario.

- Goals: The end result of most use cases. A successful diagram should describe the activities and variants used to reach the goal.

The notation for a use case diagram is pretty straightforward and doesn't involve as many types of symbols as other UML diagrams. It can use this guide to learn how to draw a use case diagram if needed a refresher. Here are all the shapes will be able to find

- Use cases: Horizontally shaped ovals that represent the different uses that a user might have.

- Actors: Stick figures that represent the people actually employing the use cases.

- Associations: A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.

- System boundary boxes: A box that sets a system scope to use cases. All use cases outside the box would be considered outside the scope of that system. For example, Psycho Killer is outside the scope of occupations in the chainsaw example found below.

- Packages: A UML shape that allows to put different elements into groups. Just as with component diagrams, these groupings are represented as file folders.

This use case diagram is a visual representation of the process required to write and publish a book. Whether an author, an agent, or a bookseller, inserting this diagram into use case scenario can help team publish the next big hit. Try this demo template to get started on own.

- Actors: The users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with application or system. They must be external objects that produce or consume data.

- System: A specific sequence of actions and interactions between actors and the system. A system may also be referred to as a scenario.

To adapt this template for any process where a customer purchases a service. With attractive colour schemes, text that’s easy to read and edit, and a wide-ranging UML shape library

The relationship between and among the actors and the use cases of Blogging system.

Super admin entity : Use cases of super admin are Manage blog, Manage blog category, Manage and create blog, Manage blog type, Manage Comments, Manage Technology blog, Manage web page, Manage users and full blogging application system.

Super user entity : Use cases of super user are Manage blog, Manage blog category, Manage and create blog, Manage blog type, Manage Comments, Manage Technology blog, Manage web page, Manage users and full blogging application system.

A use case diagram doesn't go into a lot of detail—for example, don't expect it to model the order in which steps are performed. Instead, a proper use case diagram depicts a high-level overview of the relationship between use cases, actors, and systems. Experts recommend that use case diagrams be used to supplement a more descriptive textual use case.

These internal and external agents are known as actors. So use case diagrams are consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system. So to model the entire system numbers of use case diagrams are used.

In the Unified Modelling Language (UML), a use case diagram can summarize the details of system's users (also known as actors) and their interactions with the system. To build one, use a set of specialized symbols and connectors. An effective use case diagram can help team discuss and represent.

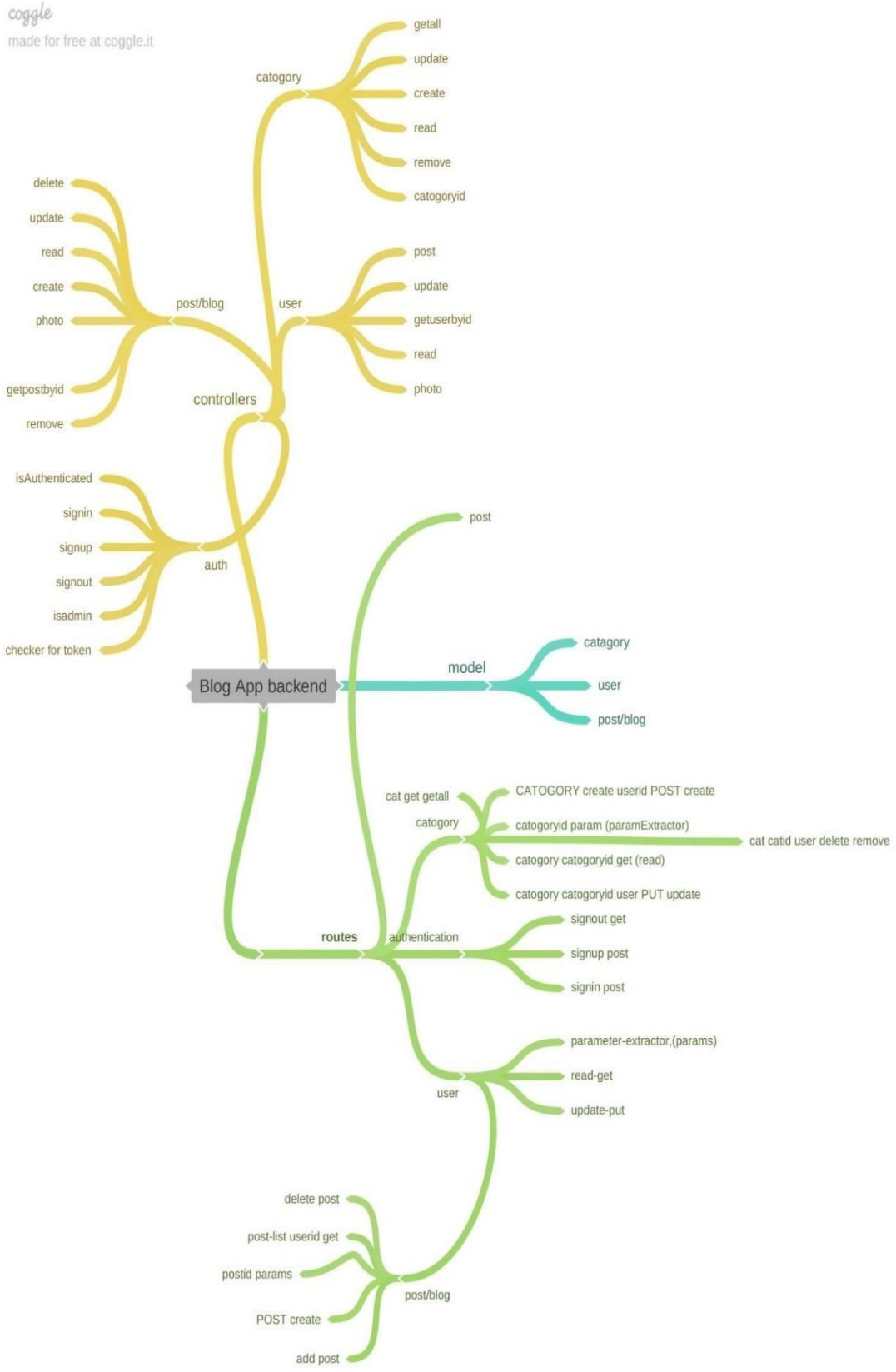


Figure 3.6.1: Use Case Diagram

3.7 Use Case Specification

Actors:

- **User**:

  i. Update Blog
  ii. Add/Delete Blog
  iii. Search Blog
  iv. Add/Edit comment

- **Admin:**

  i. Edit/Delete/Add comment

  ii Delete Blog

- **Admin**

  i. Admin login into admin panel with username and password.

  i. Admin can view user details, add new manager/delivery person/client.

  ii. Admin can add category, markets and products.

  iv. Admin can manage payouts.

  v. Admin is central authority and has all the functions.

Use Cases:

- Register.
- Login.
- Manage Profile.
- Logout.

CHAPTER 4
DESIGN

4. DESIGN

4.1 Design Goals

Design is a meaningful engineering representation of something that is to be built. It can be traced to a customer's requirements and at the same time assessed for quality against a set of predefined criteria for good design. In the software engineeringcontext, design focuses on four major areas of concern: data, architecture, interfaces, and components. The design process translate requirement into representation ofsoftware that can be accessed for a quality before core generation. Design is the process through which requirement are translated to blue print for constructing into software. Initially the blueprint depicts the holistic view of software. This is the design represented at the high level of abstraction.

During various stages of system development and design following goals havebeen setup for a complete architecture

- Analysis
- Design
- Development
- Testing
- Implementation

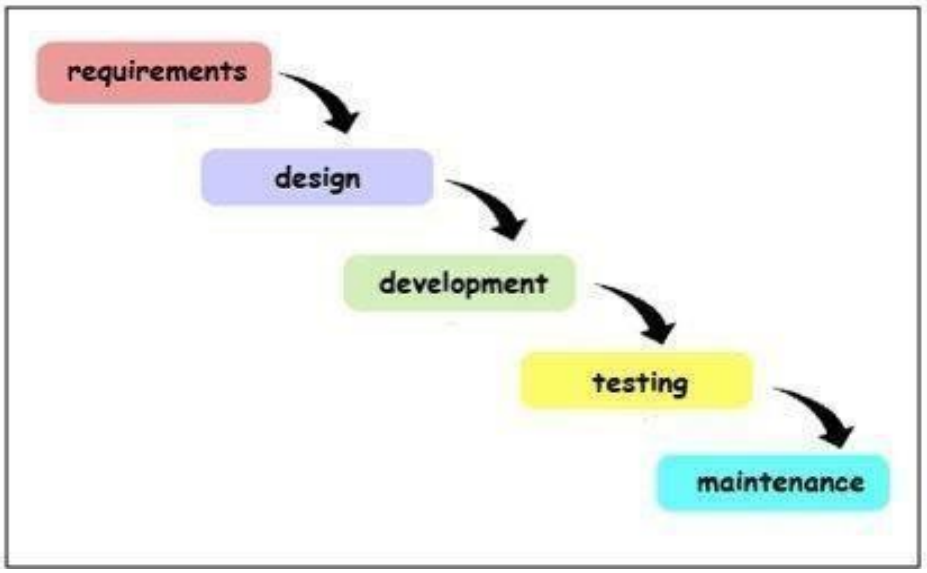


Figure 4.1.1: SDLC Cycle

The design goals of our project are quite simple. The customer satisfaction is our priority. Trying to keep the user interface simple and clean . To make this project core functionality of er commerce as guided by our sponsor like cart, authentication, social login, checkout order, view order, favourite and so on.

The Systems Development Life Cycle (SDLC) gives structure to the challenges of transitioning from the beginning to the end of the project without forgetting a step.A number of different SDLC models are used today to guide professionals through their project-based work.Here are the key pros and cons of six of the most common SDLC models for project management and leading campaigns.Waterfall is the oldest and most straightforward of the structured SDLC methodologies — finish one phase, then move on to the next. No going back.Each stage relies on information from the previous stage and has its own project plan. Waterfall is easy to understand and simple to manage.But early delays can throw off the entire project timeline.And since there is little room for revisions once a stage is completed, problems can't be fixed until get to the maintenance stage.

This model doesn't work well if flexibility is needed or if the project is long term and ongoing.The model produces ongoing releases, each with small, incremental changes from the previous release. At each iteration, the product is tested.This model emphasises interaction, as the customers, developers and testers work together throughout the project.But since this model depends heavily on customer interaction, the project can head the wrong way if the customer is not clear on the direction he or she wants to go.

4.2 Design Strategy

The design strategy of this project is to develop android application and web application module. First module of our application is android app where user can explore product, place order ,add to favourite and many more. This module contains the tracking functionality where user can track the status of order as updated by admin, manager or the driver.

The second module is the our server module where all our back-end work. In this module admin, manage, driver login respective of their role. admin can manage product, manage market. admin also has a dashboard representing count of orders, product and total

earning chart. for manager/seller can add product and can update status of order.

4.3 Architecture Diagram

Software Architecture typically refers to biggest structures of a software system, and it deals with how multiple software processes cooperate to carry out their tasks. Software Architecture is described as the organization of a system, where the system represents a set of components that accomplish the defined functions. Following figure shows architecture diagram of E-commerce system system:

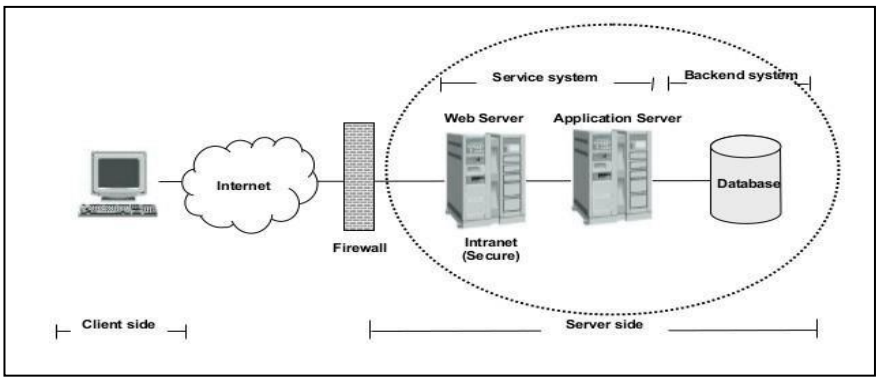


Figure 4.3.1: System Architecture Diagram

In blog app, there are different components of architecture diagram. It includes frontend, backend along with the database. In our case non-relational mongodb database is used.An "architecture" can be defined as an abstract description of entities in a system and the relationships between them. It involves a series of decision-making processes.The architecture is a structure and a vision.A "system architecture" is the embodiment of concepts and the distribution of the correspondences between the functions of things or information and formal elements. It defines the relationships among elements as well as between elements and the surrounding environment.Building a sound architecture is a complex task and a great topic for us to discuss here. After building an architecture, relevant parties must understand it and follow its dictates. An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap.A diagram much like a picture is worth a thousand words. In other words, an architectural diagram must serve several different functions. To allow relevant users to understand a system architecture and follow it in their decision-making,it need to communicate information about the architecture. Architectural diagrams provide a great way to do this. To put down some major functions, an architectural diagram needs to:

- Break down communication barriers
- Reach a consensus
- Decrease ambiguity

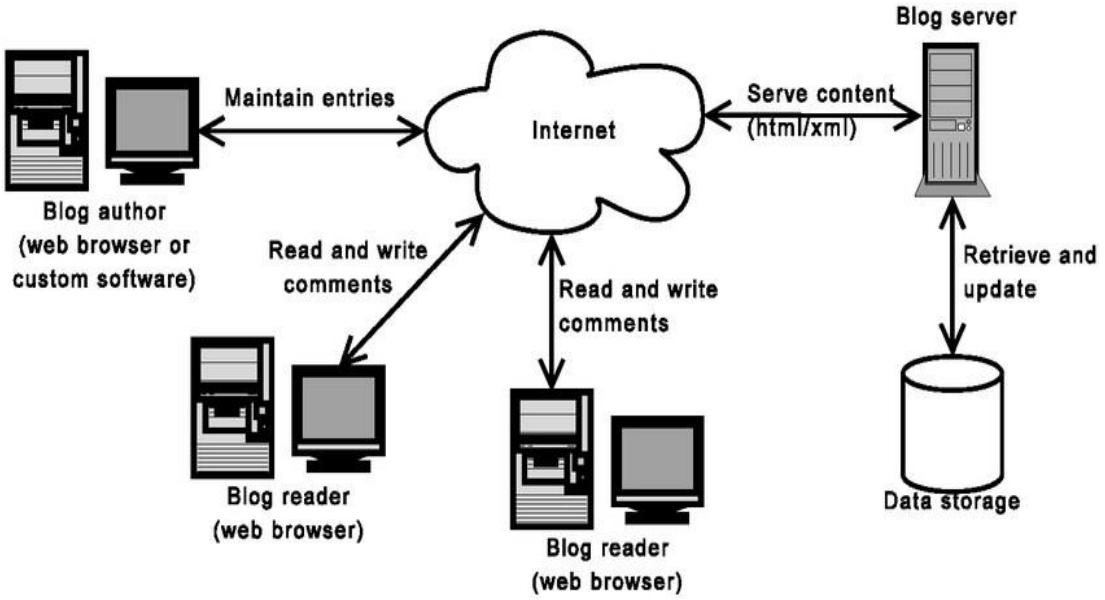


Figure 4.3.2: System Architecture Diagram

4.4 Class Diagram

The class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for Visualizing, describing and documenting different aspects of a system but also for constructing executable code ofthe software application.

The Unified Modelling Language (UML) can help model systems in various ways. One of the more popular types in UML is the class diagram. Popular among software engineers to document software architecture, class diagrams are a type of structure diagram because they describe what must be present in the system being modeled. No matter level of familiarity with UML or class diagrams, our UML software is designed to be simple and easy to use.

UML was set up as a standardized model to describe an object-oriented programming approach. Since classes are the building block of objects, class diagrams are the building blocks of UML.

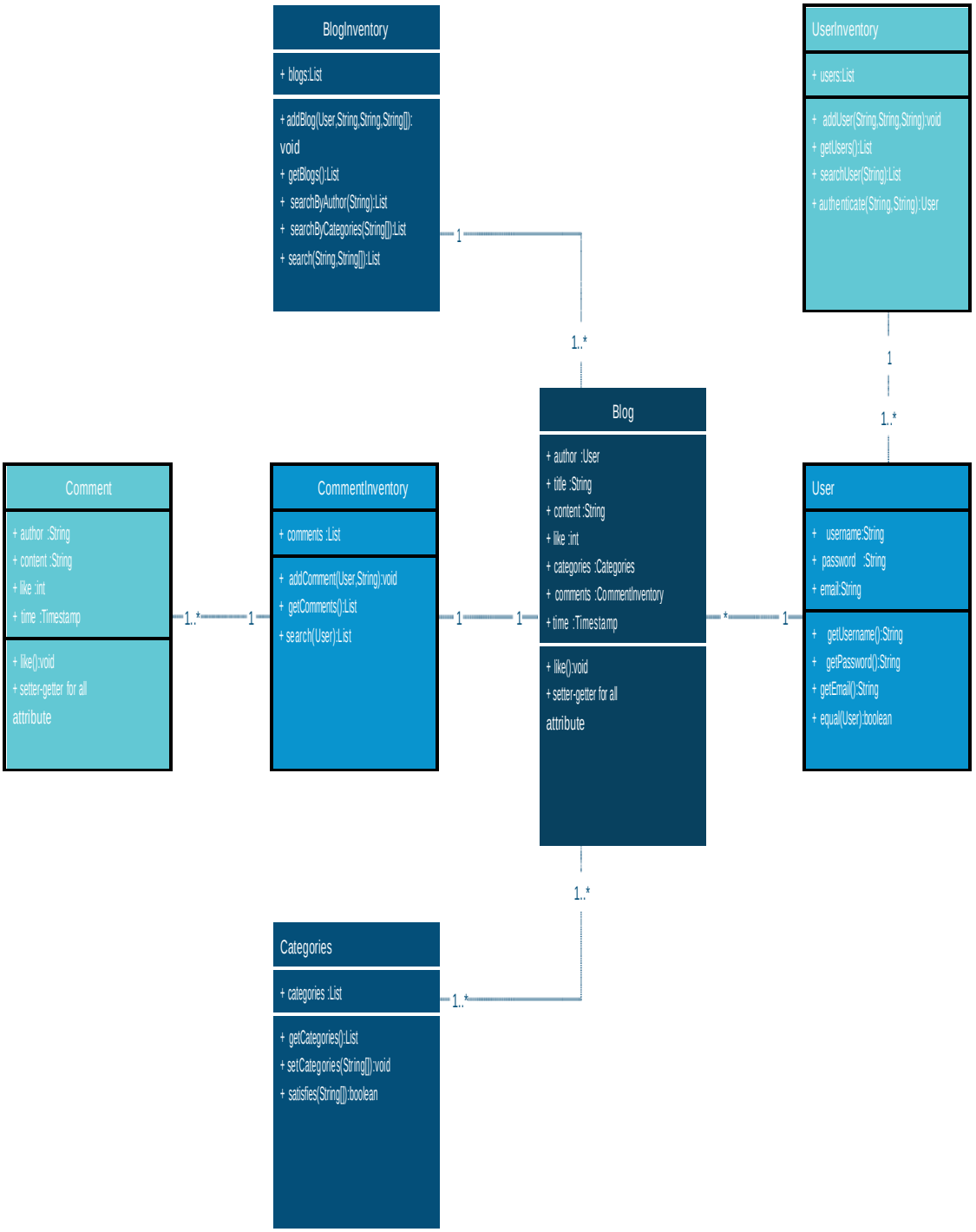


Figure 4.4.1: Class Diagram of application.

The various components in a class diagram can represent the classes that will actually be programmed, the main objects, or the interactions between classes and objects.

The class shape itself consists of a rectangle with three rows. The top row contains the name of the class, the middle row contains the attributes of the class, and the bottom section expresses the methods or operations that the class may use.

Classes and subclasses are grouped together to show the static relationship between each object.The UML shape library can help create nearly any custom class diagram using our UML diagram tool. Class diagrams offer a number of benefits for any organization. Use UML class diagrams to:

- Illustrate data models of blogs for information systems.
- Better understand the general overview of the schematics of an blog application.
- Visually express any specific needs of a blogging system and disseminate that information throughout the business.
- Create detailed charts that highlight any specific blog code needed to be programmed and implemented to the described structure.
- Provide an implementation-independent description of types used in a blogging system that are later passed between its components.

The standard class diagram of blog application is composed of three sections:

- Upper section: Contains the name of the class. This section is always required, whether are talking about the classifier or an object.
- Middle section: Contains the attributes of the class. Use this section to describe the qualities of the class. This is only required when describing a specific instance of a class.
- Bottom section: Includes class operations (methods). Displayed in list format, each operation takes up its own line. The operations describe how a class interacts with data.

There are two scopes for members: classifiers and instance for blog apps.

Classifiers are static members while instances are the specific instances of the class. If are familiar with basic OO theory, this isn't anything groundbreaking.

Depending on the context, classes in a class diagram can represent the main objects, interactions in the application, or classes to be programmed. To answer the question "What is a class diagram in UML?"

Classes: A template for creating objects and implementing behavior in a system. In UML, a class represents an object or a set of objects that share a common structure and behavior. They're represented by a rectangle that includes rows of the class name, its attributes, and its operations. When it draws a class in a class diagram, the only required to fill out the top row—the others are optional if like to provide more detail.

Interactions For Blog App

The term "interactions" refers to the various relationships and links that can exist in class and object diagrams. Some of the most common interactions include:

- Inheritance: The process of a child or sub-class taking on the functionality of a parent or superclass, also known as generalization. It's symbolized with a straight connected line with a closed arrowhead pointing towards the superclass.

In this example, the object "User" of blog would inherit all of the attributes (creator, admin, super user) of the parent class ("User") in addition to the specific attributes (model type, number of doors, auto maker) and methods of its own class (Login(), register(), logout()). Inheritance is shown in a class diagram by using a solid line with a closed, hollow arrow.

- Bidirectional association: The default relationship between two classes. Both classes are aware of each other and their relationship with the other. This association is represented by a straight line between two classes.

In the example above, the User class and Admin class are interrelated. At one end of the line, and takes on the association of "assignedRole" with the multiplicity value of 0..1, so when the instance of user exists, it can either have one instance of User associated with it or no points associated with it. In this case, a separate Caravan class with a multiplicity value of 0..* is needed to demonstrate that a Blog could have multiple instances of User associated with it. Since one user instance could have multiple "createBlog" associations—in other words, one user could go on multiple road trips—the multiplicity value is set to 0..*

- Unidirectional association: A slightly less common relationship between two classes. One class is aware of the other and interacts with it. Unidirectional association is modeled with a straight connecting line that points an open arrowhead from the knowing class to the known class.

As an example, on creating health blog, it might run across a speed trap where a it is not relevent, but not knowing about it until getting a notification in the mail. It isn't drawn in the image, but in this case, the multiplicity value would be 0..* depending on how many times toxic words are found.

A class diagram can show the relationships between each object in a blog management system, including creator information, admin responsibilities, and blog toxicity. The example below provides a useful overview of the hotel management system. Get started on a class diagram by clicking the template below.

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.

Following entity relationship diagram shows the blog database system. Such data models are typically used in the first phase of information system design; for example, they are used to describe information requirements and/or the types of information to be stored in the database during the requirements analysis phase. This template is an ER diagram of the blog database for reference.

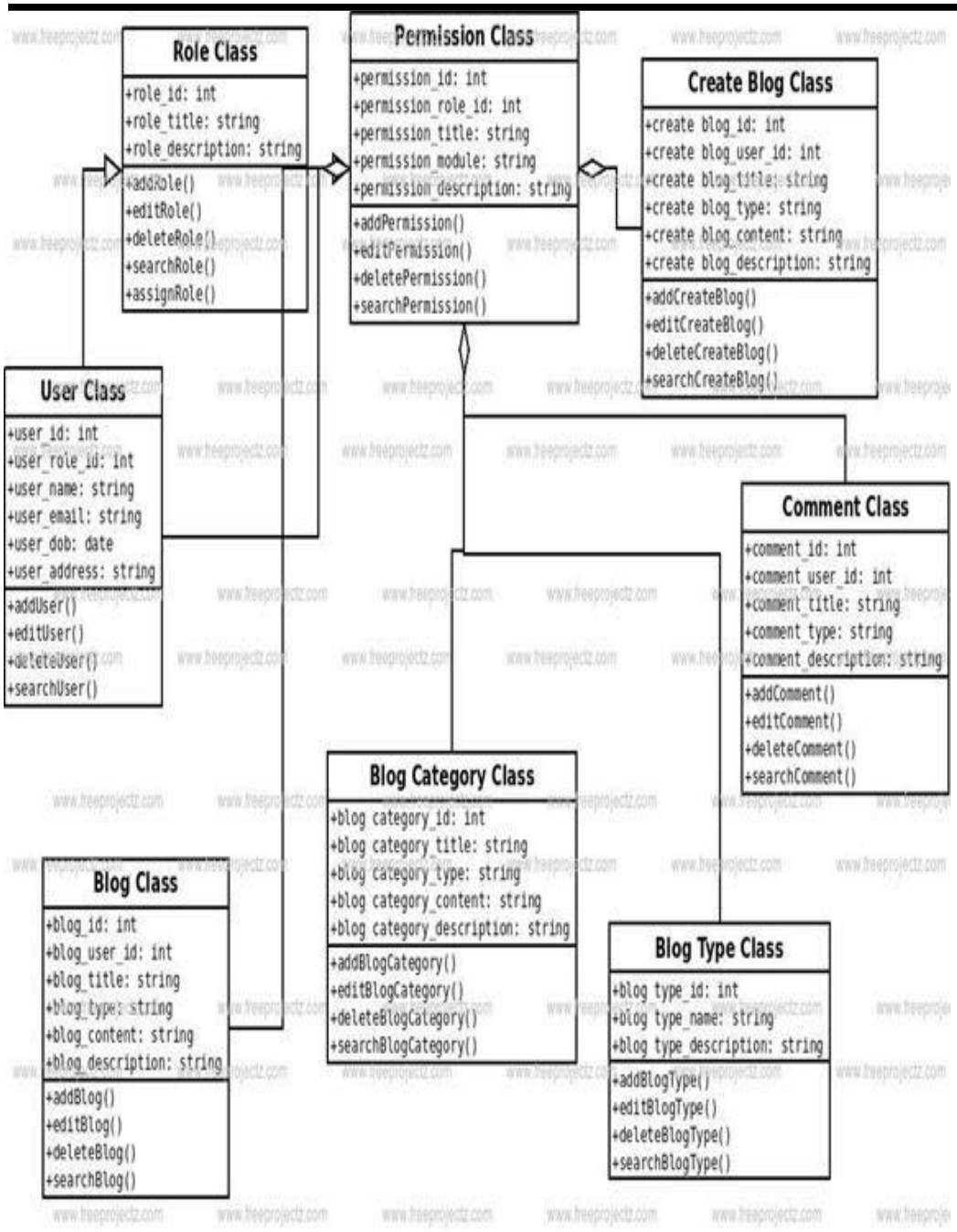


Figure 4.4.2: Entity relationship diagram of application.

4.5 Sequence Diagram

UML Sequence diagrams are used to show how objects interact in a given situation An important Characteristics of a sequence diagram is that time passes from top to bottom: the interaction starts near the top of the diagram and ends at the bottom (i.e. Lower equals later.) A popular use for them is to document the dynamics in an - object-oriented system for each Key Collaboration diagrams are created that show how objects interact in various representative scenarios for that collaboration.

Sequence diagram commonly contain the following:

- Objects in following diagram
- Links in following diagram
- Messages in following diagram

Following figures shows the sequence diagram of project:

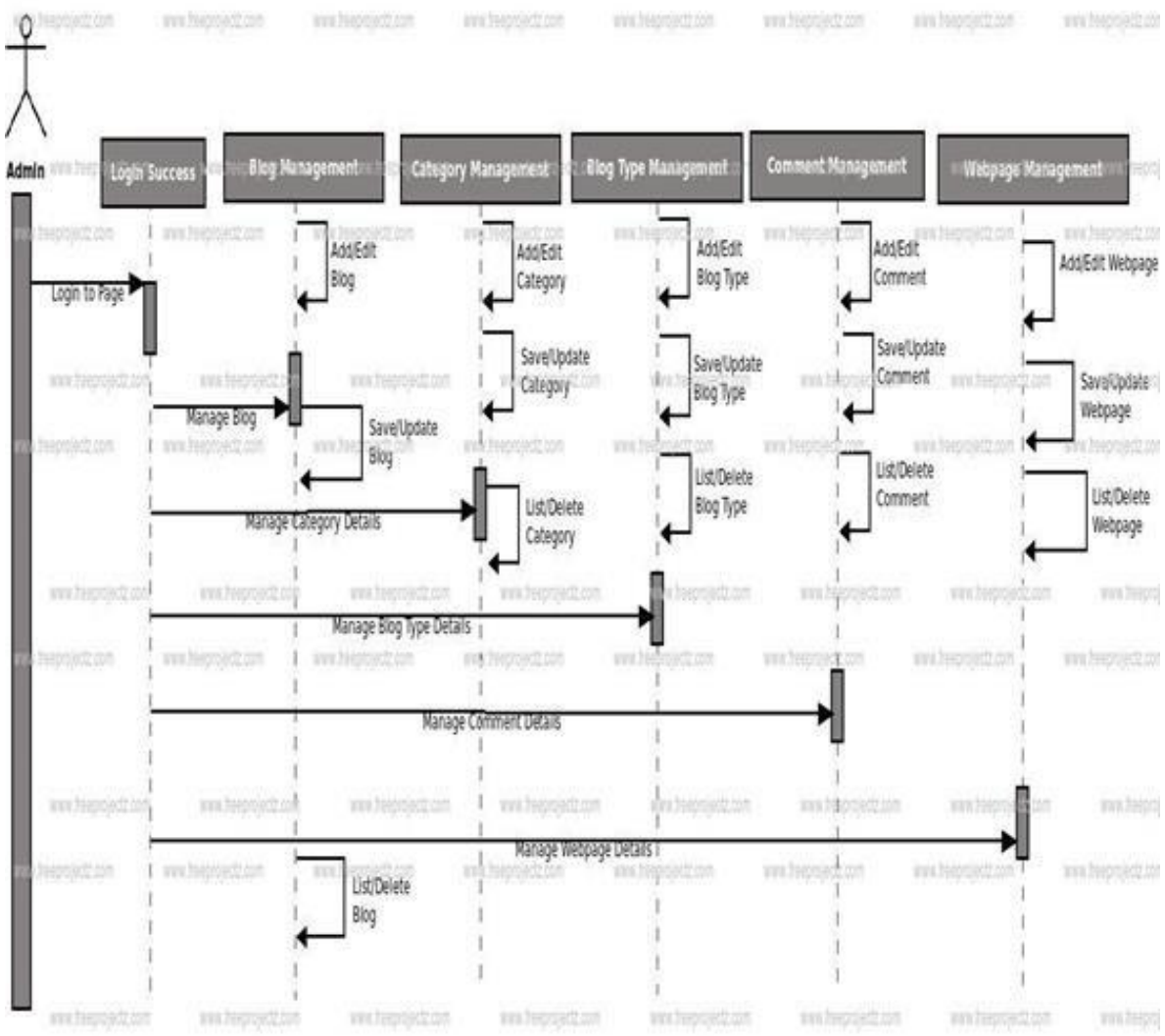


Figure 4.5.1: Sequence Diagram of Admin and User.

To understand what a sequence diagram for admin and user is, it's important to know the role of the Unified Modeling Language, better known as UML. UML is a modeling toolkit that guides the creation and notation of many types of diagrams, including behavior diagrams, interaction diagrams, and structure diagrams.

In blogging, sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.

Note that there are two types of sequence diagram for bloggings: UML diagrams and code-based diagrams. The latter is sourced from programming code and will not be covered in this guide. UML diagramming software is equipped with all the shapes and features it will need to model both.

Sequence diagrams for blog app can be useful references for businesses and other organizations. Try drawing a sequence diagram to:

- Represent the blog creator details of a UML use case.
- Model the logic of a sophisticated procedure, function, or operation for articles.
- For blogging see how objects and components interact.
- Plan and understand the detailed functionality of blog users.

The following scenarios are ideal for using a sequence diagram for blog app:

- Blog usage scenario: Scenario is a diagram of how system could potentially be used
- ML sequence diagram of blog to explore the logic of a use case and can use it to explore the logic of any function, procedure, or complex process.
- Service logic: If it is considered a service to be a high-level method used by different clients, a sequence diagram is an ideal way to map that out.
- Sequence diagram Visio - Any sequence diagram that create with Visio can also be uploaded into .vsd and .vdx file import and is a great Microsoft Visio alternative. Almost all of the images to see in the UML section of this site were generated.

To understand what a sequence diagram is, it is important to be familiar with its symbols and components. Sequence diagrams are made up of the following icons and elements:

Use the following arrows and message symbols to show how information is transmitted between objects. These symbols may reflect the start and execution of an operation or the sending and reception of a signal.

Social media is progressively exposed to issues of damaging conduct, for example, private assaults and cyberbullying. It is troublesome and time consuming to manually check which of the comments need to be blocked. Therefore, automating the process of identifying the abusive comments and blocking them would not only save time, but also provide safety to the users.

The elegance with online comment posting has grown at a faster rate and Social media is progressively exposed to issues of damaging conduct, for example, private assaults and cyber bullying.

A sequence diagram of blog app is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.

A deployment diagram for blog app in the Unified Modeling Language models the physical deployment of artifacts on nodes. To describe a website, for example, a deployment diagram would show what hardware components ("nodes") exist(e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected. The nodes appear as boxes, and theartifacts allocated to each node appear as rectangles within the boxes. Nodes may have subnodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

There are two types of Nodes:

- Device Node
- Execution Environment Node

Device nodes are physical computing resources with processing memory and services to execute software, such as typical computers or mobile phones. An execution environment node (EEN) is a software computing resource that runs within an outer node and which itself provides a service to host and execute other executable software elements.

4.6 Activity Diagram

Activity diagram for blog app is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deals with all type of flow control by using different elements like fork, join etc.

Following figures shows the activity diagram of project:

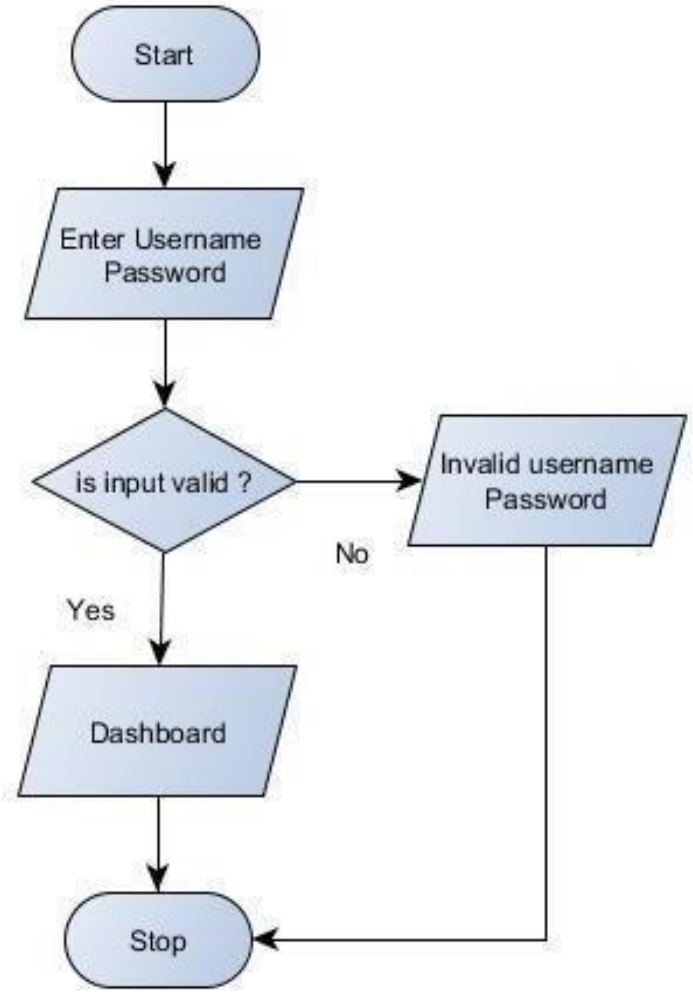


Figure 4.6.1: Activity Diagram of User side.

The Unified Modeling Language includes several subsets of diagrams, including structure diagrams, interaction diagrams, and behavior diagrams. Activity diagrams, along with use case and state machine diagrams, are considered behavior diagrams because they describe what must happen in the system being modeled.

Stakeholders have many issues to manage, so it's important to communicate with clarity and brevity. Activity diagrams help people on the business and development sides of an organization come together to understand the same process and behavior. Using a set of specialized symbols—including those used for starting, ending, merging, or receiving steps in the flow—to make an activity diagram, which covers in more depth within this activity diagram guide.

Activity diagrams present a number of benefits to users. Consider creating an activity diagram to:

- Demonstrate the logic of an algorithm to implement blog app.
- Describe the steps performed in a UML use case to create blog app.
- Illustrate a business process or workflow between users and the system.
- Simplify and improve any process by clarifying complicated use cases.
- Model software architecture elements, such as method, function, and operation.

Before beginning making an activity diagram, first understand its makeup. Some of the most common components of an activity diagram include:

- Action in blog app: A step in the activity wherein the users or software perform a given task. Actions are symbolized with round-edged rectangles.
- Decision node of blog app: A conditional branch in the flow that is represented by a diamond. It includes a single input and two or more outputs.
- Control flows of blog app: Another name for the connectors that show the flow between steps in the diagram.
- Start node of blog app: Symbolizes the beginning of the activity. The start node is represented by a black circle.

- End node of blog app: Represents the final step in the activity. The end node is represented by an outlined black circle.

These activity diagram shapes and symbols are some of the most common types that will find in UML diagrams.

Activity diagrams map out process flows in a way that's easy to understand. Consider the two examples below when it comes to creating UML activity diagrams.

Activity diagram for a login page

Many of the activities people want to accomplish online—checking email, managing finances, ordering clothes, etc.—require them to log into a website. This activity diagram shows the process of logging into a website, from entering a username and password to successfully logging in to the system. It uses different container shapes for activities, decisions, and notes. The ideal tool for creating any kind of UML flowchart, whether it’s an activity diagram, a use case diagram, or a component diagram. It offers in-editor collaboration tools and instant web publishing so can demonstrate the functionality of system to others.

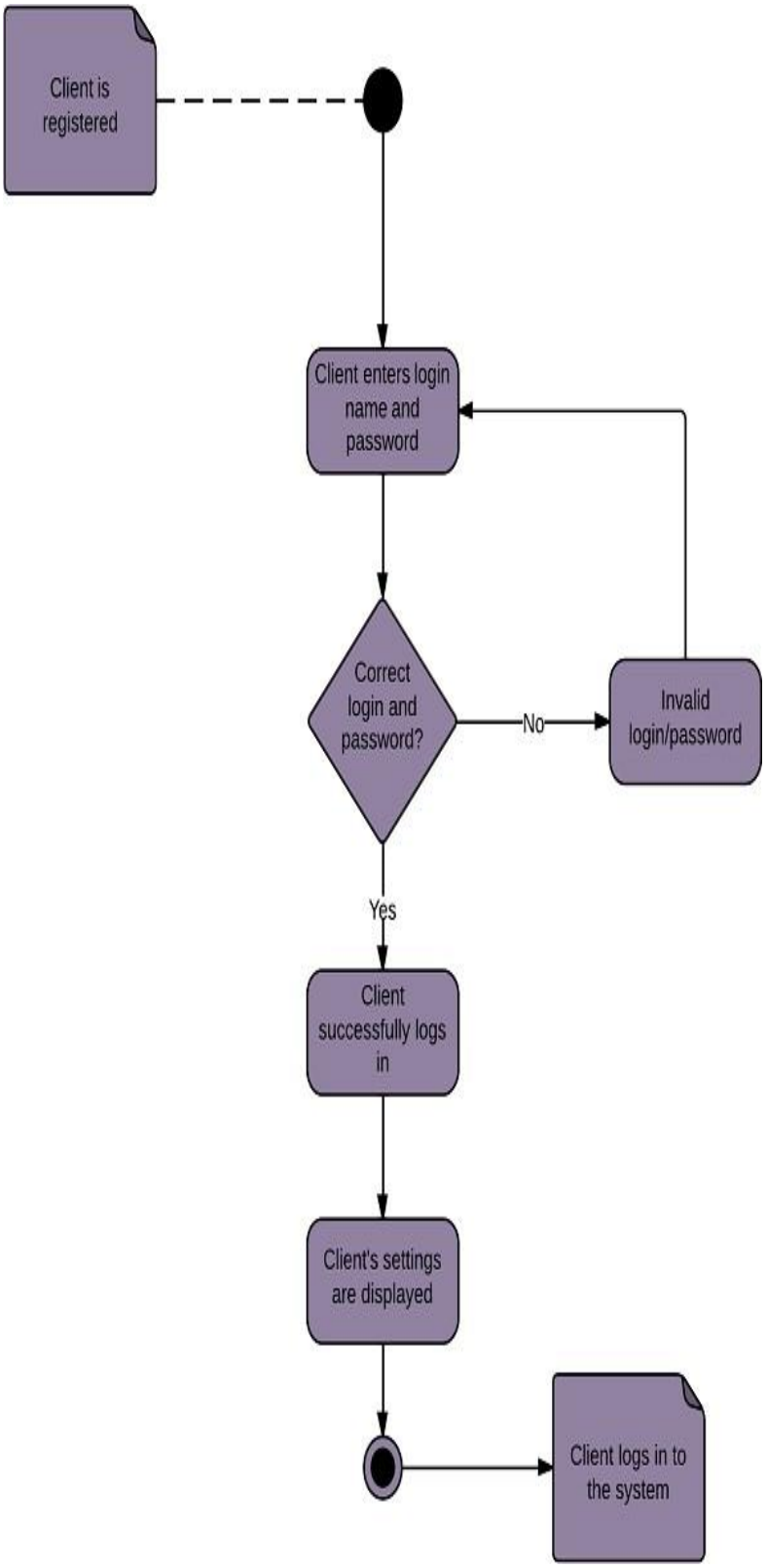


Figure 4.6.2: Activity Diagram of User login.

CHAPTER 5
IMPLEMENTATION

5. IMPLEMENTATION

5.1 Implementation Strategy

The explosion of social media is changing the way everyone communicates. In case of blogging platforms the current problems are like not having a clear topic and niche for the blog, not keeping on topic, absence of feedback or comments, inability to handle engagement, inability to handle offensive comments.

So, the social media app will try to deal with these kind of issuesProject Component:
It consists of two units:

1) Admin :

This is a web application developed in php using html, CSS and JavaScript .Thisunit of application has admin and manager level of functionality. In this unit user can add product, manage orders, manage users, manage markets and so on.

2) User :

This is a mobile application developed using flutter. flutter is a hybrid mobile application development platform written in dart programming language. User unithas functionality of exploring product, and placing orders.

➢ User Module:

The flow chart in the figure 5.1.1 is for user where a valid user can login for the client.

Before that user need to complete his profile user need to add delivery detail and payment time and then proceed to order. User can order product through delivery or pick up mode and on confirming order it will be placed and all detail will be saved into our database. User can track order from track order page.If not registered, user can register and then login. User can explore product through search, filters and by categories, on tap on product user navigated to product detail page where user can see all the detail of product side if he has already registered. If not registered, user can register and then login.User can explore product through search, filters and by categories, on tap on product user navigated to product detail page where user can see all the detail of product, clickon add to cart from product detail page and product will be added to cart. On cart pageuser can increase or

decrease the quantity of product and the final step is checkout where user can see the final list of product before placing order.

Before that user need to complete his profile user need to add delivery detail and payment time and then proceed to order. User can order product through delivery or pick up mode and on confirming order it will be placed and all detail will be saved into our database. User can track order from track order page.

➢ Implementation level details

To make the front-end of the website React library is used. There are several libraries that need to integrate with React:

1. React Router Dom: This library is used for navigation
2. React Hook Form: This library is used to easily maintain the form of our app.
3. Stripe: Stripe API is integrated to maintain the payment info. It is a suite of payment APIs that powers commerce for online businesses of all sizes.
4. React-ga: This is a JavaScript module that can be used to include Google Analytics tracking code in a website or app that uses React for its front-end codebase.
5. Sass: Sass is a CSS preprocessor library. It is a powerful tool whichenables us to create much cleaner and more maintainable styling for our websites.
6. Bootstrap: It is CSS framework for responsivedesign.
7. Netlify: To deploy our frontend application.
8. Tenserflow-toxicity-model : To detect toxic comments.
9. Summarizer : To extract the summary from the text.

Backend Application

For developing the backend use the following technologies:

1. Node.js: Node.js is a free, open-sourced, cross-platform run-time environment for JavaScript that lets developers write command-line tools and server-side scripts outside of a browser.

2. Express.js: Express is a minimal and flexible Node.js web application framework that provides us a robust set of features to develop web and mobile applications. It facilitates the rapid development of Node-based Web applications.

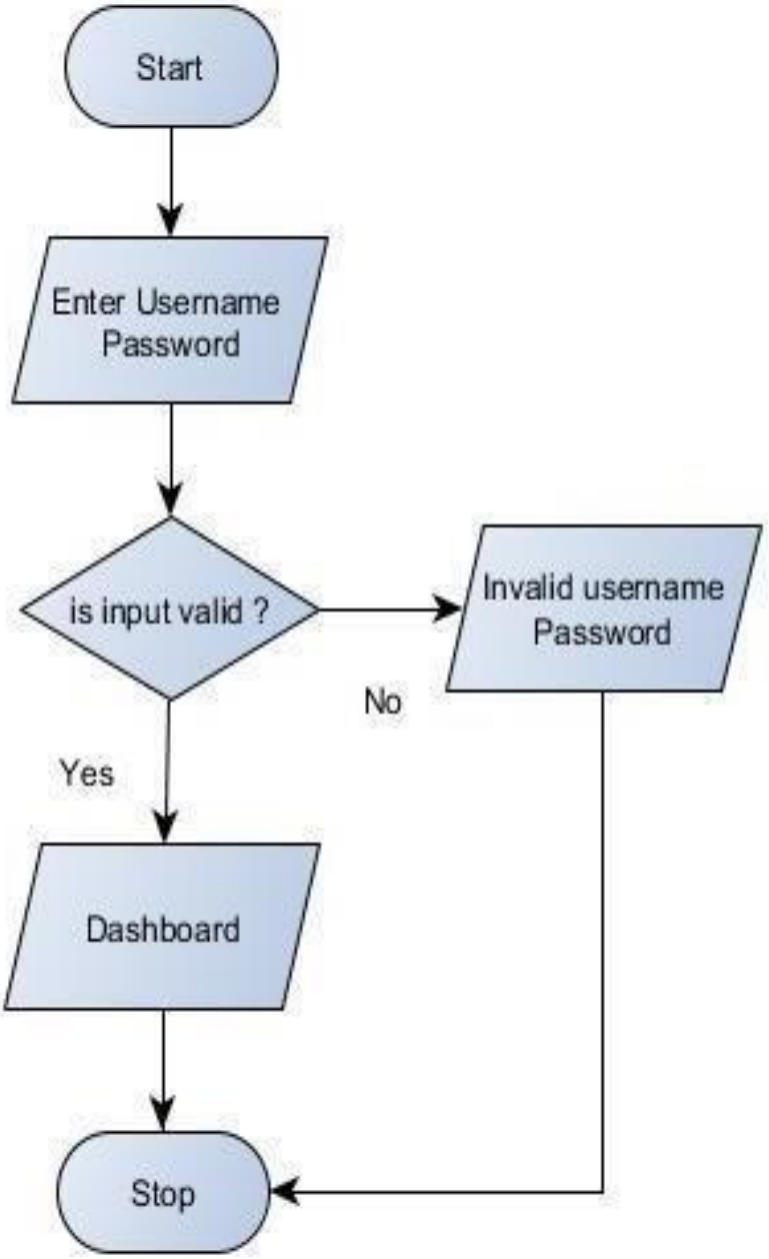


Figure 5.1.1: User Flowchart

On cart pageuser can increase or decrease the quantity of product and the final step is checkout where user can see the final list of product before placing order. Before that user need to complete his profile user need to add delivery detail and payment time and then proceed to order. User can order product through delivery or pick up mode and on confirming order it will be placed and all detail will be saved into our database. User can track order from track order page.

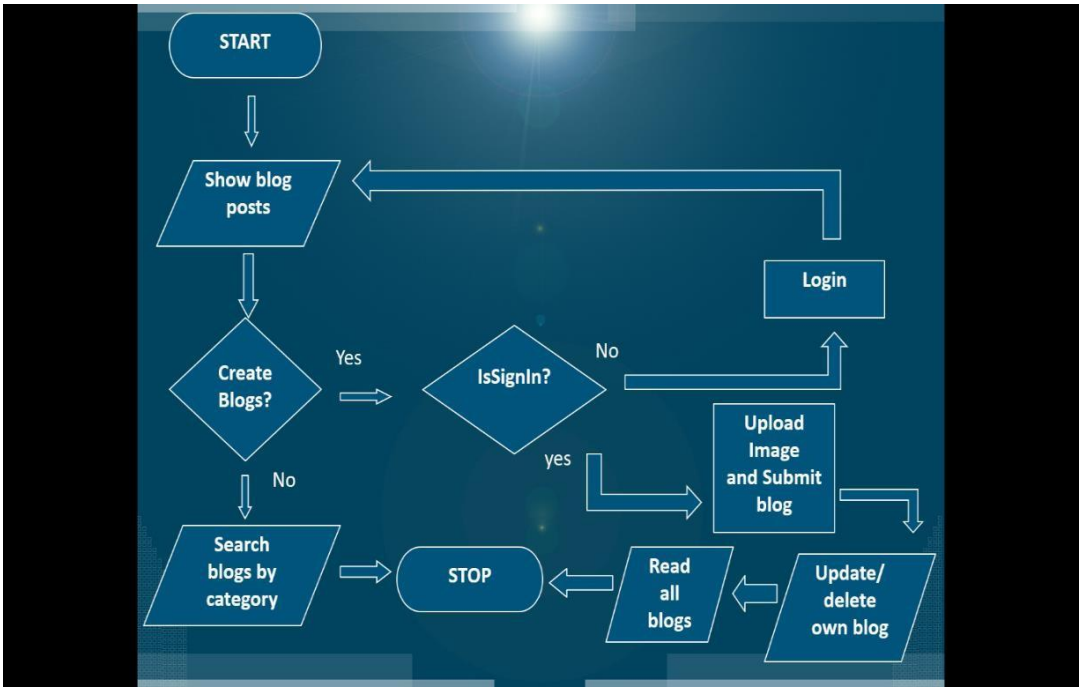


Figure 5.1.2: Blog app Flowchart

5.2 Hardware Platform Used

The Hardware Requirement may serve as the basis for a contract for the implementation of the system and should therefore be complete and consistent in specification. The minimum hardware requirement of the system is mentioned below.

PROCESSOR : Intel® Core™ i3-4005U CPU@ 1.70 GHz 1.70GHz

RAM : 4.00 GB

Hard Disk : 20 GB

It should be noted that the better the hardware facilities available, the higher the responsetime of the system.

5.3 Software Platform Used

5.3.1 Windows OS

This Computer Operating System (OS) developed by Microsoft Corporation to run personal computers (PCs). Featuring the first graphical user interface (GUI) for IBM-compatible PCs, the Windows OS soon dominated the PC market. Here Windows 10 OS is used to run the project.

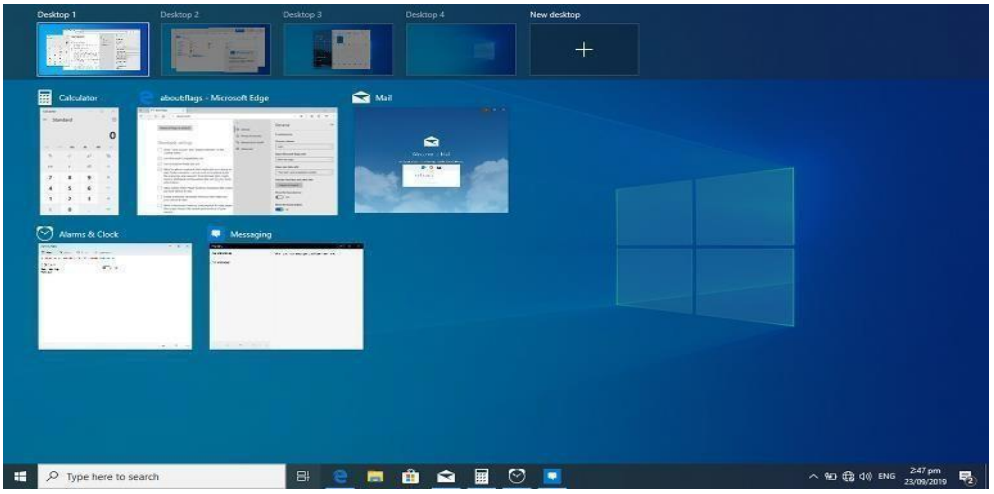


Figure 5.3.1.1: Windows OS

A new iteration of the Start menu is used on the Windows 10 desktop, with a list of places and other options on the left side, and tiles representing applications on the right. The menu can be resized, and expanded into a full-screen display, which is the default option inTablet mode. A new virtual desktop system was added. A feature known as Task View displays all open windows and allows users to switch between them, or switch between multiple workspaces. Universal apps, which previously could be used only in full-screen mode, can now be used in self-contained windows similarly to other programs. Program windows can now be snapped to quadrants of the screen by dragging them to the corner. When a window is snapped to one side of the screen, Task View appears and the user is prompted to choose a second window to fill the unused side of the screen (called "Snap Assist"). Windows' system icons were also changed.

5.3.2 Visual Studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store, and Microsoft Silverlight. It can produce both native codes and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that expand the functionality at almost every level—including adding support for source control systems (like Subversion and Git) and adding new tool sets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Azure DevOps client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML, and CSS. Support for other languages such as Python,[9] Ruby, Node.js, and M among others is available via plug-ins.

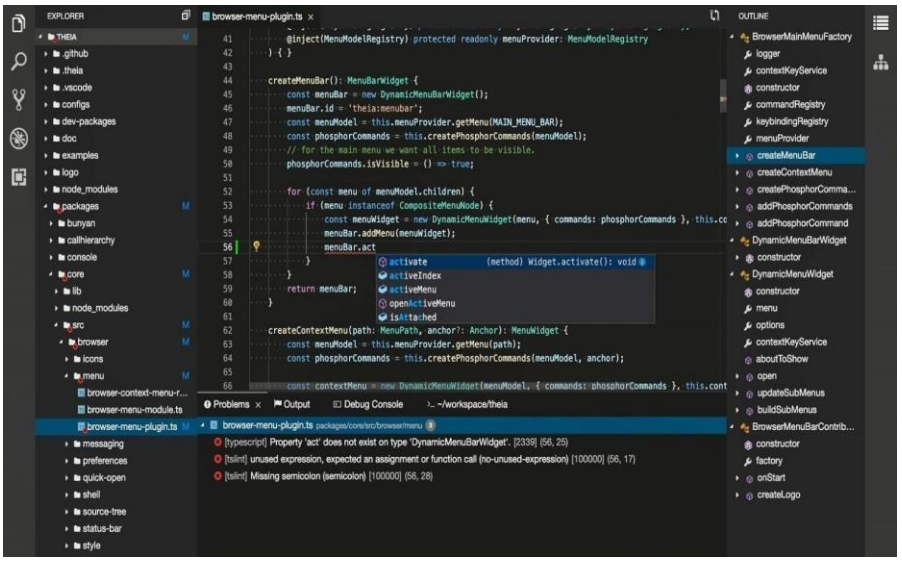


Figure 5.3.2.1: Visual Studio

5.3.3 Node JS

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command-line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web application development around a single programming language, rather than different languages for server-side and client-side scripts.

Though .js is the standard filename extension for JavaScript code, the name "Node.js" doesn't refer to a particular file in this context and is merely the name of the product. Node.js has an event-driven architecture capable of asynchronous I/O. These design choices aim to optimize throughput and scalability in web applications with many input/output operations, as well as for real-time Web applications (e.g., real-time communication programs and browser games).

When Node.js performs an I/O operation, like reading from the network, accessing a database or the filesystem, instead of blocking the thread and wasting CPU cycles waiting, Node.js will resume the operations when the response comes back.

This allows Node.js to handle thousands of concurrent connections with a single server without introducing the burden of managing thread concurrency, which could be a significant source of bugs.

Node.js has a unique advantage because millions of frontend developers that write JavaScript for the browser are now able to write the server-side code in addition tothe client-side code without the need to learn a completely different language.

In Node.js the new ECMAScript standards can be used without problems, it doesn't have to wait for all users to update their browsers - in charge of deciding which ECMAScript version to use by changing the Node.js version, and it can also enable

Node.js represents a "JavaScript everywhere" paradigm, unifying web application development around a single programming language, rather than different languages for server-side and client-side scripts.

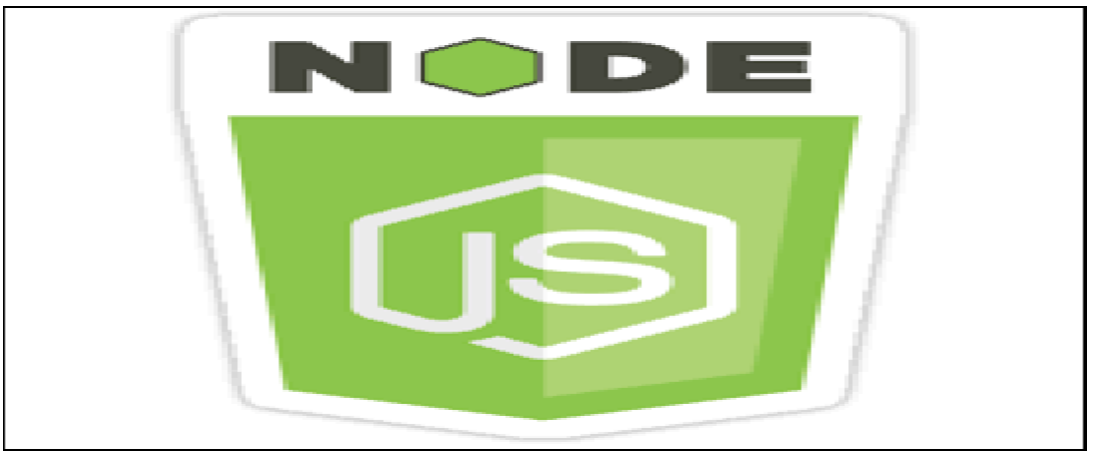


Figure 5.3.3.1: Node js

specific experimental features by running Node.js with flags.

Node.js is an open-source and cross-platform JavaScript runtime environment. It is a popular tool for almost any kind of project. Node.js runs the V8 JavaScript engine, the core of Google Chrome, outside of the browser. This allows Node.js to be very performant.A Node.js app runs in a single process, without creating a new thread for every request. Node.js provides a set of asynchronous I/O primitives in its standard library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm.When Node.js performs an I/O operation, like reading from the network, accessing a database or the filesystem, instead of blocking the thread and wasting CPU cycles waiting, Node.js will resume the operations when the response comes back.This allows Node.js to handle thousands of concurrent connections with a single server without introducing the burden of managing thread concurrency, which could be a significant source of bugs.Node.js has a unique advantage because millions of frontend developers that write JavaScript for the browser are now able to write the server-side code in addition to the client-side code without the need to learn a completely different language.

In Node.js the new ECMAScript standards can be used without problems, as one don't have to wait for all users to update their browsers – That person is in charge of deciding which ECMAScript version to use by changing the Node.js version, and one can also enable specific experimental features by running Node.js with flags.

npm with its simple structure helped the ecosystem of Node.js proliferate, and now the npm registry hosts over 1,000,000 open source packages one can freely use.

5.3.4 MongoDB, React and Express

MongoDB is an open-source document database that provides high performance, high availability, and automatic scaling. In simple words, it can be said that - Mongo DB is a document-oriented database. MongoDB, the most popular NoSQL database, is an open-source document-oriented database. The term 'NoSQL' means 'non-relational. It means that MongoDB isn't based on the table-like relational database structure but provides an altogether different mechanism for the storage and retrieval of data. This format of storage is called BSON (similar to JSON format).

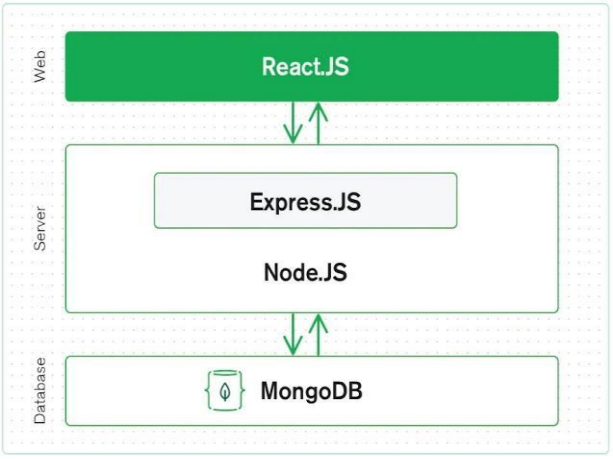


Figure 5.3.3.4: MongoDB

MongoDB Atlas is MongoDB's fully-managed cloud database service. The service is built to handle enterprise workloads, with support for global clusters.

One can store the data with Amazon Web Services (AWS), Google Cloud Platform, or Microsoft Azure. However, one don't need to set up an account with any of these platforms. MongoDB Atlas takes care of all this behind the scenes.

MongoDB Atlas also automatically handles backend administrative processes such as provisioning resources, setting up clusters, or scaling services. Most of the tasks one perform are simple point-and-click operations that one carry out through the service's centralized web interface.

What is included in the MongoDB Atlas free cluster?

No introduction to MongoDB Atlas is complete without mentioning its free M0 tier, which includes:

- 512 MB of storage
- Shared RAM
- Highly available replica sets, end-to-end encryption, automated patches, REST API
- Max connections: 100
- Network performance: Low
- Max databases: 100

The free tier cluster regions available are:

- N. Virginia (us-east-1)
- Frankfurt (eu-central-1)
- Singapore (ap-southeast-1)
- Mumbai (ap-south-1)

Reactjs

When React 16.8 was released officially in early February 2019, it shipped with an additional API that lets one use state and other features in React without writing a class. This additional API is called Hooks and they're becoming popular in the React ecosystem, from open-sourced projects to being used in production applications.

React Hooks are completely opt-in which means that rewriting existing code is unecessary, they do not contain any breaking changes, and they're available for use with the release of

React 16.8. Some curious developers have been making use of the Hooks API even before it was released officially, but back then it was not stable and was only an experimental feature. Now it is stable and recommended for React developers to use.

What Are React Hooks? #

React Hooks are in-built functions that allow React developers to use state and lifecycle methods inside functional components, they also work together with existing code, so they can easily be adopted into a codebase. The way Hooks were pitched to the public was that they allow developers to use state in functional components but under the hood, Hooks are much more powerful than that. They allow React Developers to enjoy the following benefits:

- Improved code reuse;
- Better code composition;
- Better defaults;
- Sharing non-visual logic with the use of custom hooks;
- Flexibility in moving up and down the components tree.

With React Hooks, developers get the power to use functional components for almost everything they need to do from just rendering UI to also handling state and also logic — which is pretty neat.

Motivation Behind The Release Of React Hooks #

According to the ReactJS official documentation, the following are the motivation behind the release of React Hooks:

- Reusing stateful logic between components is difficult.

With Hooks, one can reuse logic between components without changing their architecture or structure.

- Complex components can be difficult to understand.

When components become larger and carry out many operations, it becomes difficult to understand in the long run. Hooks solve this by allowing one to separate a particular single

component into various smaller functions based upon what pieces of this separated component are related (such as setting up a subscription or fetching data), rather than having to force a split based on lifecycle methods.

- Classes are quite confusing.

Classes are a hindrance to learning React properly; one would need to understand how this in JavaScript works which differs from other languages. React Hooks solves this problem by allowing developers to use the best of React features without having to use classes.

The Rules Of Hooks #

There are two main rules that are strictly to be adhered to as stated by the React core team in which they outlined in the hooks proposal documentation.

- Make sure to not use Hooks inside loops, conditions, or nested functions;
- Only use Hooks from inside React Functions.

Basic React Hooks #

There are 10 in-built hooks that was shipped with React 16.8 but the basic (commonly used) hooks include:

- useState()
- useEffect()
- useContext()
- useReducer()

These are the 4 basic hooks that are commonly used by React developers that have adopted React Hooks into their codebases.

The useState() hook allows React developers to update, handle and manipulate state inside functional components without needing to convert it to a class component. Let's use the code snippet below is a simple Age counter component and it will use it to explain the power and syntax of the useState() hook.

If one has noticed, our component looks pretty simple, concise and it's now a functional component and also does not have the level of complexity that a class component would have.

The useState() hook receives an initial state as an argument and then returns, by making use of array destructuring in JavaScript, the two variables in the array can be named what. The first variable is the actual state, while the second variable is a function that is meant for updating the state by providing a new state.

This is how our component should look when it is rendered in our React application. By clicking on the "Increase my Age" button, the state of the age will change and the component would work just like a class component with state.

useEffect() #

The useEffect() hook accepts a function that would contain effectual code. In functional components, effects like mutations, subscriptions, timers, logging, and other effects are not allowed to be placed inside a functional component because doing so would lead to a lot of inconsistencies when the UI is rendered and also confusing bugs.

In using the useEffect() hook, the effectual function passed into it will execute right after the render has been displayed on the screen. Effects are basically peeked into the imperative way of building UIs that is quite different from React's functional way.

By default, effects are executed mainly after the render has been completed, but one has the option to also fire them when certain values change.

The useEffect() hook mostly into play for side-effects that are usually used for interactions with the Browser/DOM API or external API-like data fetching or subscriptions. Familiarity with how React lifecycle methods work, one can also think of useEffect() hook as component mounting, updating and unmounting — all combined in one function. It lets us replicate the lifecycle methods in functional components.

Expressjs

Node.js is a JavaScript run time environment which is used to create server-side applications and tools. Node.js is fast, portable, and written in JavaScript but it does not directly support common tasks such as handling requests, serving files, and handling HTTP methods such as GET and POST. This is where Node.js's rich ecosystem comes to our aid.

Express.js (Express) is a light web framework which sits on top of Node.js and it adds functionality like (middleware, routing, etc.) and simplicity to Node.js.

When creating a Node.js web application,writing a single JavaScript application which listens to requests from the browser, based on the request, the function will send back some data or an HTML web page.

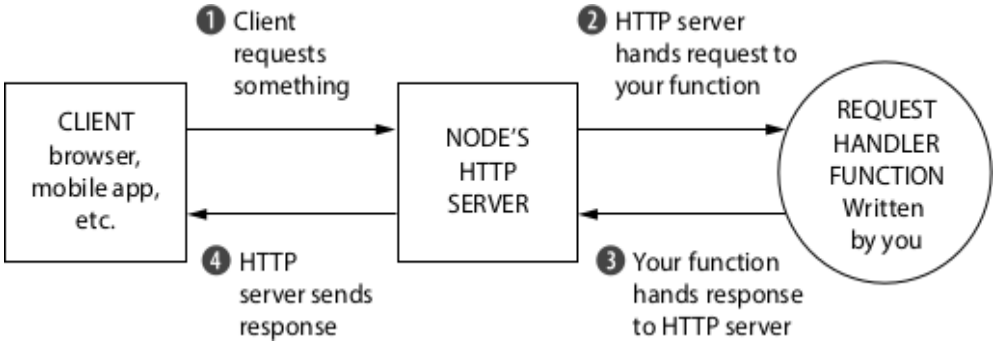


Figure 5.3.3.5: Express js

Figure of Expressjs Architecture is shown above

A request handler is a JavaScript function which takes a request and sends an appropriate response.

Node.js APIs can get complex and writing how to handle a single request can end up being over 50 lines of code. Express makes it easier to write Node.js web applications.

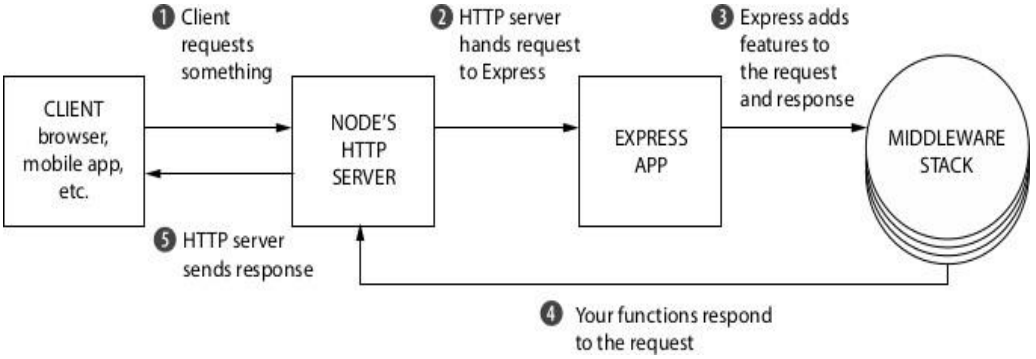


Figure 5.3.3.6: MongoDB

Figure Expressjs Middleware Architecture

Advantages of using Express with Node.js

- Express lets take away a lot of the complexities of Node.js while adding helpful functions to a Node.js HTTP server.
- Instead of a large request handler function, Express allows us to handle requests by writing many small modular and maintainable functions.
- Express is not opinionated, meaning Express does not enforce any "right way" of doing things. Any compatible middleware can be used, and it is possible to structure the app as needed, making it flexible.
- One can integrate with a template rendering engine (also called a view rendering engine in some articles) of our choice like Jade, Pug, EJS, etc.

A template engine enables to use static template files and at runtime change the values of variables in those files.

- "middleware" set up for request processing.

Basic Express Application

Let's create a basic Express example app. To use Express,one first need to install it via npm using the command below.Next, let's write the code for our example app. Create a

file called app.js. The code above creates a basic Express application. To run this script, go to command prompt and enter the command node app.js in the project directory. Let's look at what the code above is doing.The first line imports the express module. The second line creates an Express application by calling the top-level express() function.

In the console, one can see Application started and Listening on port 3000 and if visit http://localhost:3000/ one can see HELLO WORLD.

Our app variable (express application) has methods for handling requests and configuring how the application behaves. One can create multiple apps this way, each with their own requests and responses. Lets examine the code in section two. app.get() is a function, called route definition, which tells the express app how to handle an HTTP GET request to our server.

This function takes two main parameters, the first is the route or path which is the relative path from the root of the server; the second is a function that is invoked whenever there is a request to that path. In this case, listening for GET requests to / which is the root of the website. The second parameter, the callback function, has two arguments req and res. req represents the request sent from the browser to the server. res represents the response that the server sends back. The code in section three starts a server on the port 3000. Responses can be seen by going to localhost:3000

➢ Core Parts of Express

Middleware

Middleware is a set of functions that sit between a raw request and the final intended route. Middleware functions have access to all the HTTP requests coming to the server. Middleware can handle tasks such as logging, sending static files, authorization, and session management, etc.

In Node.js, the request and response objects are passed to one function (request handler) that is written, in Express these objects are passed through a set of functions, called the middleware stack.Express will start at the first function in the stack and execute in order down the stack. Every function in the stack takes three arguments request, response and next. next is a function, that when called Express executes the next function in the stack. This is a subtle difference between middleware and

a route handler which is seen above. Let's look at a basic static file server to understand middleware. Initialize a new npm project. Then create a directory named static and copy-paste any available static files into the folder (text, images, etc.).

Execute the following commands in the terminal. The touch command creates an empty file.

Our app will have a logger function and a static file serving function.If running this file using node app.js and go to localhost:3000/dummy_file.txt, and can be seen on the screen file1.If going to the URL localhost:3000, one can see an error Cannot GET / because it did not configure a route handler for that path. Let's look at the code.The logger logs every request that comes into the server. app.use is used to define a middleware function, it takes a function.

The next() function call tells Express to move onto the next function in the stack (remove the next() call in script, it can be noticed that it takes forever for the page to load, this is because the request gets stuck on this middleware function).Using the path module to join the relative URL (from the request) and the directory name.

The fs module provides an API for interacting with the file system. Checking if the file exists, if it does not, going to next function in the stack if it does then will return that file using res.sendFile. One can write our own middleware functions or import them similar to how to import our modules in Node.js using require.

Install it using npm. One can call the use() on the Express app object to add the middleware to the stack.Express comes with express.static middleware bundled with it, it can be used to serve static files instead of the function in the previous section. It provides better security and performance than the function to be written. JavaScript app.use(express.static("static"); //relative pathAny requested files in the directory "static" are served. localhost:3000/dummy_file.txt will show the same result as above.

One can call static() multiple times to use multiple static asset directories. For example, consider that have two directories static and public with static files and it wrote the following code:JavaScript app.use(express.static("static"); app.use(express.static("public");

Making a request like localhost:3000/hello.html, Express looks up the files in the static directory then public directory if the file exists then returns hello.html.

Routing

Express makes request handling easier by mapping requests to different request handlers. A request handler is a function which handles all the requests to a specific path with a specific HTTP method. In the basic example above, seeing how to handle a GET request. As an application grows in size the routes grow as well as do the request handlers. Lets see how one can use Routers to split a large app into smaller, maintainable functions.According to the documentation, a Router is "an isolated instance of middleware and routes. Routers can be thought of as "mini" applications only capable of performing middleware and routing".

Routers can be used like middleware functions, they can be added to middleware stack using the app.use() function. A simple example:Create a folder called routes and a file called api_router.js inside it. When starting the app and visit the URL localhost:3000/api/route1, here Success!! Message can be seen. Take a look at all the router functions here.

Grabbing route parameters. Suppose while building a website for a company that showcases their products, each product has a productID.URL is required for product 1 to be /product/1.Instead of defining a route for every product, single route can be defined for everything in the form of product/productID and then return a file based on the productID. Here's a rough example below that shows modification for use case. Using Regular Expressions to match routesRegular Expressions (RE) are patterns used to match character combinations in strings. One can use RE to match parameters and define our routes.For example, using the example above, if it is wanted the productId to be only an integer one can try it.

Template Engines

Websites are built with HTML, can dynamically generate HTML pages using Express. Dynamically generated HTML pages are useful when it is necessary to show real time data or change a page's details based on the user.

A template engine allows the use of static template files and at runtime replace variables in a template file with actual data.There are different template engines available like Pug, Jade, and EJS. Let's see a basic template using EJS.First let's install it using npm. Type npm install ejs and then create a directory called views to store the templates and HTML files.Create a file called index.ejs in the views directory. Going to the webpage that can be seen Hello and Welcome !!!. It is possible to write JavaScript expressions inside the <%= exp %>. Look at the docs for the complete syntax and rules.

Testing Express Applications: Testing is an important part of developing software. Read this article where it is discussed testing Node.js applications using Mocha and Chai.

5.4 Testing

Testing is a process, which reveals errors in the program. It is the majorquality measure employed during software development. During testing, the program is executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as it is expected to perform.

A. Testing Strategies

In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at differing phases of software development are:

a) Unit Testing:

Unit Testing is done on individual modules as they are completed and become executable. It is confined only to the designer's requirements. Unit tests are handy for verifying the behavior of a single function, method, or class. The test package provides the core framework for writing unit tests, and the flutter test package provides additionalutilities for testing widgets.

b) Integrating Testing:

Integration tests work as a pair: first, deploy an instrumented application to a real device or emulator and then "drive" the application from a separate test suite, checking to make sure everything is correct along the way. To create this test pair, usethe flutter driver package. It provides tools to create instrumented apps and drivethose apps from a test suite.

c) System Testing:

It involves in-house testing of the entire system before delivery to the user. Its aim is to satisfy the user the system meets all requirements of the client's specifications.

d) Acceptance Testing:

It is a pre-delivery testing in which entire system is tested at client's site on real world data to find errors.

Black box testing is a type of software testing in which the functionality of the software is not known. The testing is done without the internal knowledge of the products.

Black box testing can be done in following ways:

1. Syntax Driven Testing – This type of testing is applied to systems that can be syntactically represented by some language. For example- compilers,language that can be represented by context free grammar. In this, the test cases are generated so that each grammar rule is used at least once.

2. Equivalence partitioning – It is often seen that many type of inputs work similarly so instead of giving all of them separately and can group them together and test only one input of each group. The idea is to partition the input domain of the system into a number of equivalence classes such that each member of class works in a similar way, i.e., if a test case in one class results in some error, other members of class would also result into same error. The technique involves two steps:

1. Identification of equivalence class – Partition any input domain into minimum two sets: valid values and invalid values. For example, if the valid range is 0 to 100 then select one valid input like 49 and one invalid like 104.
2. Generating test cases –

(i) To each valid and invalid class of input assign unique identification number.
(ii) Write test case covering all valid and invalid test case considering that no two invalid inputs mask each other.

To calculate the square root of a number, the equivalence classes will be:
(a) Valid inputs:

- Whole number which is a perfect square- output will be an integer.
- Whole number which is not a perfect square- output will be decimal number.
- Positive decimals

(b) Invalid inputs:

- Negative numbers(integer or decimal).
- Characters other that numbers like “a”,”!”,”;”,etc.

3. Boundary value analysis – Boundaries are very good places for errors to occur. Hence if test cases are designed for boundary values of input domain then the efficiency of testing improves and probability of finding errors also increase. For example – If valid range is 10 to 100 then test for 10,100 also apart from valid and invalid inputs.

4. Cause effect Graphing – This technique establishes relationship between logical input called causes with corresponding actions called effect. The causes and effects are represented using Boolean graphs. The following steps are followed:

1. Identify inputs (causes) and outputs (effect).
2. Develop cause effect graph.
3. Transform the graph into decision table.
4. Convert decision table rules to test cases.

5. Requirement based testing – It includes validating the requirements given in SRS of software system.

6. Compatibility testing – The test case result not only depend on product but also infrastructure for delivering functionality. When the infrastructure parameters are changed it is still expected to work properly. Some parameters that generally affect compatibility of software are:

1. Processor (Pentium 3,Pentium 4) and number of processors.
2. Architecture and characteristic of machine (32 bit or 64 bit).
3. Back-end components such as database servers.
4. Operating System (Windows, Linux, etc).

White box testing techniques analyze the internal structures the used data structures, internal design, code structure and the working of the software rather than just the functionality as in black box testing. It is also called glass box testing or clear box testing or structural testing.

Working process of white box testing:

- Input: Requirements, Functional specifications, design documents, source code.
- Processing: Performing risk analysis for guiding through the entire process.

- Proper test planning: Designing test cases so as to cover entire code. Execute rinse-repeat until error-free software is reached. Also, the results are communicated.

- Output: Preparing final report of the entire testing process.

Testing techniques:

- Statement coverage: In this technique, the aim is to traverse all statement at least once. Hence, each line of code is tested. In case of a flowchart, every node must be traversed at least once. Since all lines of code are covered, helps in pointing out faulty code.

Branch Coverage: In this technique, test cases are designed so that each branch from all decision points are traversed at least once. In a flowchart, all edges must be traversed at least once.

- Condition Coverage: In this technique, all individual conditions must be covered as shown in the following example:

0. READ X, Y
1. IF(X == 0 || Y == 0)
2. PRINT '0'

In this example, there are 2 conditions: X == 0 and Y == 0. Now, test these conditions get TRUE and FALSE as their values. One possible example would be:

3. #TC1 – X = 0, Y = 55
4. #TC2 – X = 5, Y = 0

- Multiple Condition Coverage: In this technique, all the possible combinations of the possible outcomes of conditions are tested at least once. Let's consider the following example:

0. READ X, Y
1. IF(X == 0 || Y == 0)
2. PRINT '0'
3. #TC1: X = 0, Y = 0
4. #TC2: X = 0, Y = 5
5. #TC3: X = 55, Y = 0
6. #TC4: X = 55, Y = 5

Hence, four test cases required for two individual conditions. Similarly, if there are n conditions then 2n test cases would be required.

- Basis Path Testing: In this technique, control flow graphs are made from code or flowchart and then Cyclomatic complexity is calculated which defines the number of independent paths so that the minimal number of test cases can be designed for each independent..

Steps:

0. Make the corresponding control flow graph
1. Calculate the cyclomatic complexity
2. Find the independent paths
3. Design test cases corresponding to each independent path

Flow graph notation: It is a directed graph consisting of nodes and edges. Each node represents a sequence of statements, or a decision point. A predicate node is the one that represents a decision point that contains a condition after which the graph splits. Regions are bounded by nodes and edges.

Cyclomatic Complexity: It is a measure of the logical complexity of the software and is used to define the number of independent paths. For a graph G, V(G) is its cyclomatic complexity.

Calculating V(G):

1. V(G) = P + 1, where P is the number of predicate nodes in the flow graph
2. V(G) = E – N + 2, where E is the number of edges and N is the total number of nodes
3. V(G) = Number of non-overlapping regions in the graph

- V(G) = 4 (Using any of the above formulae)
  No of independent paths = 4

- #P1: 1 – 2 – 4 – 7 – 8
- #P2: 1 – 2 – 3 – 5 – 7 – 8
- #P3: 1 – 2 – 3 – 6 – 7 – 8
- #P4: 1 – 2 – 4 – 7 – 1 – . . . – 7 – 8
- Loop Testing: Loops are widely used and these are fundamental to many algorithms hence, their testing is very important. Errors often occur at the beginnings and ends of loops.
  - Simple loops: For simple loops of size n, test cases are designed that:
- Skip the loop entirely
- Only one pass through the loop
- 2 passes
- m passes, where m < n
- n-1 ans n+1 passes

- Nested loops: For nested loops, all the loops are set to their minimum count and it starts from the innermost loop. Simple loop tests are conducted for the innermost loop and this is worked outwards till all the loops have been tested.
- Concatenated loops: Independent loops, one after another. Simple loop tests are applied for each.

  If they're not independent, treat them like nesting.

Advantages:

1. White box testing is very thorough as the entire code and structures are tested.
2. It results in the optimization of code removing error and helps in removing extra lines of code.
3. It can start at an earlier stage as it doesn't require any interface as in case of black box testing.
4. Easy to automate.

Disadvantages:

1. Main disadvantage is that it is very expensive.
2. Redesign of code and rewriting code needs test cases to be written again.
3. Testers are required to have in-depth knowledge of the code and programming language as opposed to black box testing.

CHAPTER 6
CONCLUSION

6. CONCLUSION

The amount of data available online is limitless. Think about a normal college student, who has to go through thousands of pages of documents each semester. That is why text summarization is necessary.The main purpose of text summarization is to get the most precise and useful information from a large document and eliminate the irrelevant or less important ones.Text summarization can be done either manually, which is time-consuming, or through machine algorithms and AIs, which takes very little time and is a better option.Once The data in the database can be added additional reports and improve the reports visualization with more graphics. By developing additional report exporters and formatters, companies may have customized progress reports in various formats based on tasks, roles, and time.Nowadays witnesses of the creation of huge software systems with the participation of many employees. There is a need for moderncommunication channels for all the project participants for improving the inter-communication process. One way to do it is to build a messagingsystem so that the users are able to send and receive all kinds of messages such as emails, instant messages, SMS/MMS messages, and so on.The application is inherently open, portable, and scalable because enterprise technology is used for its development. It provides small and medium enterprises with plenty of project tracking functionalities with easy maintenance at a low cost.

CHAPTER 7
FUTURE WORK

7. FUTURE WORK

- Detection of toxic blog articles.

  Online media aim for reaching ever bigger audience and for attracting ever longer attention span. This competition creates an environment that rewards sensational, fake, and toxic news. To help limit their spread and impact, with propose to and develop a news toxicity detector that can recognize various types of toxic content.

- Checking if image is matching with blog content

  It is very necessary to check if the image uploaded by user is matching with the blog content he has uploaded otherwise it can create unnecessary confusion.

- Taking feedback from users

  User feedback is information collected directly from users/customers about their reactions to a product, service, or website experience. This feedback is collected with a variety of tools, such as Customer Satisfaction (CSAT) or Net Promoter Score (NPS) surveys.

- Implementation of good design pattern

  In software engineering, a design pattern is a general repeatable solution to a commonly occurring problem in software design. A design pattern isn't a finished design that can be transformed directly into code. It is a description or template for how to solve a problem that can be used in many different situations.

In software engineering, a design pattern is a general repeatable solution to a commonly occurring problem in software design. A design pattern isn't a finished design that can be transformed directly into code. It is a description or template for how to solve a problem that can be used in many different situations.

Uses of Design Patterns

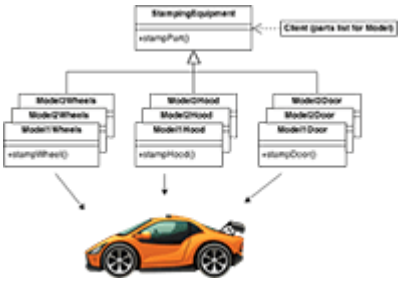
Design patterns can speed up the development process by providing tested, proven development paradigms. Effective software design requires considering issues that may not become visible until later in the implementation. Reusing design patterns helps to prevent subtle issues that can cause major problems and improves code readability.

Often, people only understand how to apply certain software design techniques to certain problems. These techniques are difficult to apply to a broader range of problems. Design patterns provide general solutions, documented in a format that doesn't require specifics tied to a particular problem.

In addition, patterns allow developers to communicate using well-known, well understood names for software interactions. Common design patterns can be improved over time, making them more robust than ad-hoc designs.

Creational design patterns

These design patterns are all about class instantiation. This pattern can be further divided into class-creation patterns and object-creational patterns. While class-creation patterns use inheritance effectively in the instantiation process, object-creation patterns use delegation effectively to get the job done.



- Abstract Factory

Creates an instance of several families of classes

- Builder

Separates object construction from its representation

- Factory Method

Creates an instance of several derived classes

- Object Pool

Avoid expensive acquisition and release of resources by recycling objects that are no longer in use

- Prototype

A fully initialized instance to be copied or cloned

Structural design patterns

These design patterns are all about Class and Object composition. Structural class-creation patterns use inheritance to compose interfaces. Structural object-patterns define ways to compose objects to obtain new functionality.

- Adapter

Match interfaces of different classes

- Bridge

Separates an object's interface from its implementation

- Composite

A tree structure of simple and composite objects

- Decorator

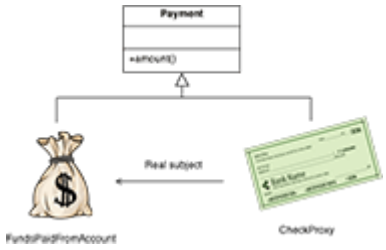
Add responsibilities to objects dynamically

- Facade

A single class that represents an entire subsystem

- Flyweight

A fine-grained instance used for efficient sharing

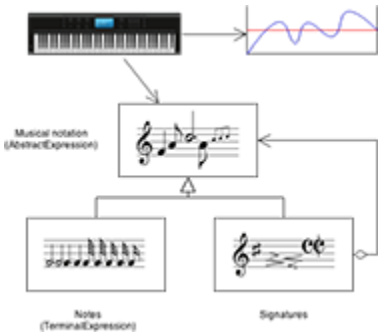


Private Class Data

Restricts accessor/mutator access

Behavioral design patterns

These design patterns are all about Class's objects communication. Behavioral patterns are those patterns that are most specifically concerned with communication between objects.



- Chain of responsibility

A way of passing a request between a chain of objects

- Command

Encapsulate a command request as an object

- Interpreter

A way to include language elements in a program

- Iterator

Sequentially access the elements of a collection

- Mediator

Defines simplified communication between classes

- Memento

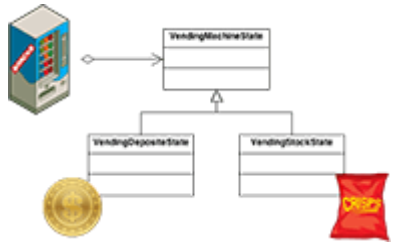
Capture and restore an object's internal state

- Null Object

Designed to act as a default value of an object

- Observer

A way of notifying change to a number of classes



- Template method

Defer the exact steps of an algorithm to a subclass

- Visitor

Defines a new operation to a class without change

- Removing duplicate elements from database schema

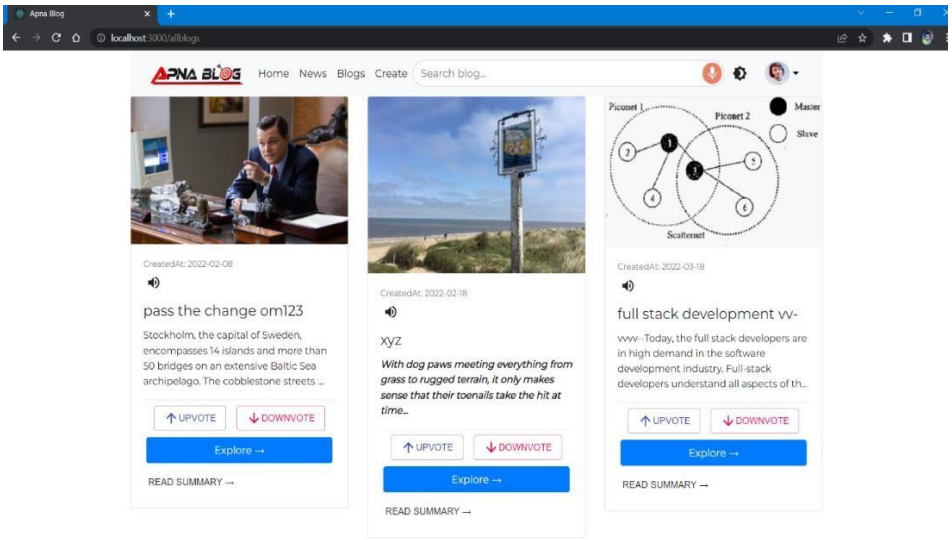
  Data redundancy is a condition created within a database or data storage technology in which the same piece of data is held in two separate places.

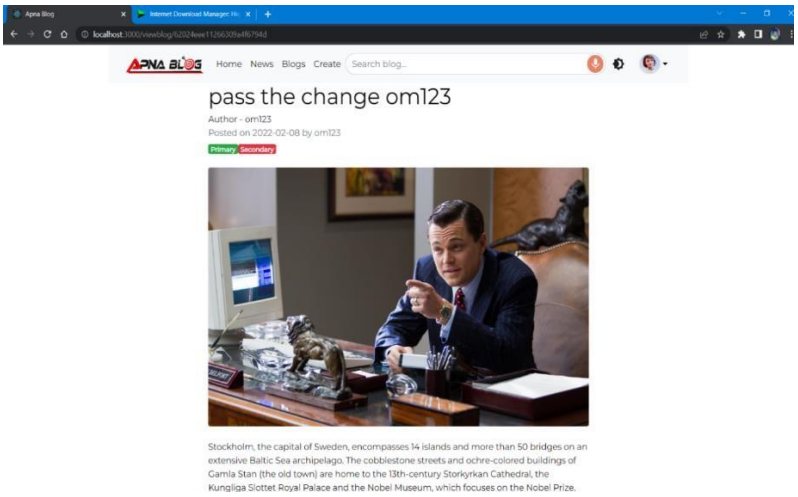
USER MANUAL

Admin Manual

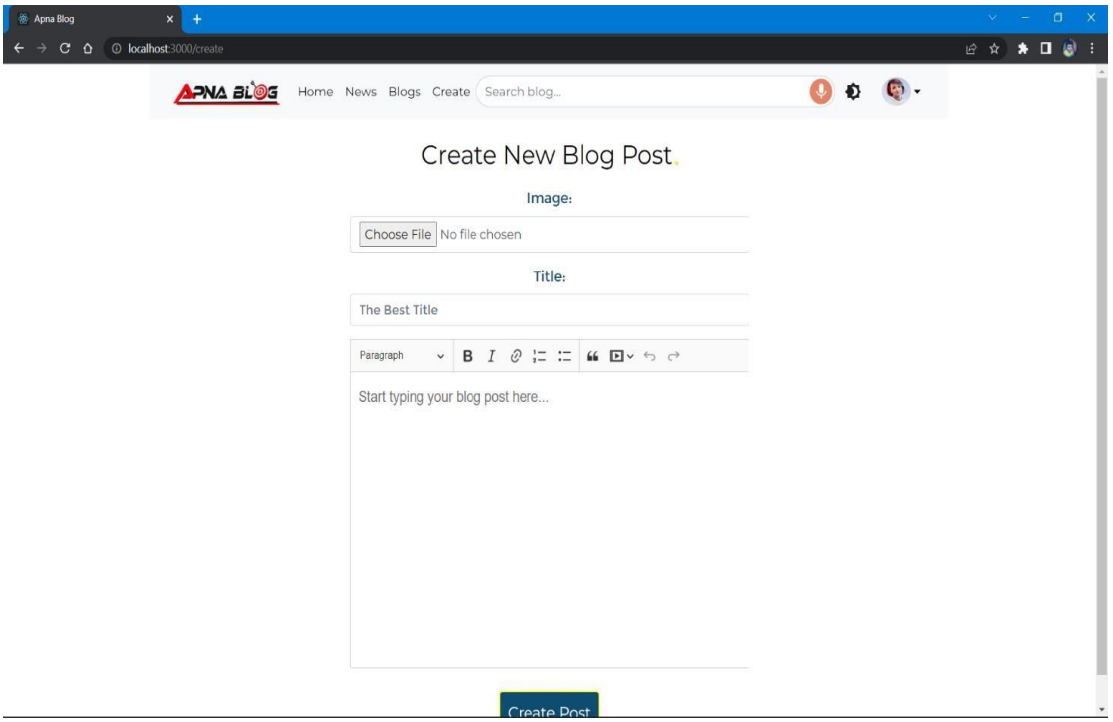
To have a VISUAL STUDIO IDE in our system or Follow the following steps to run the application:

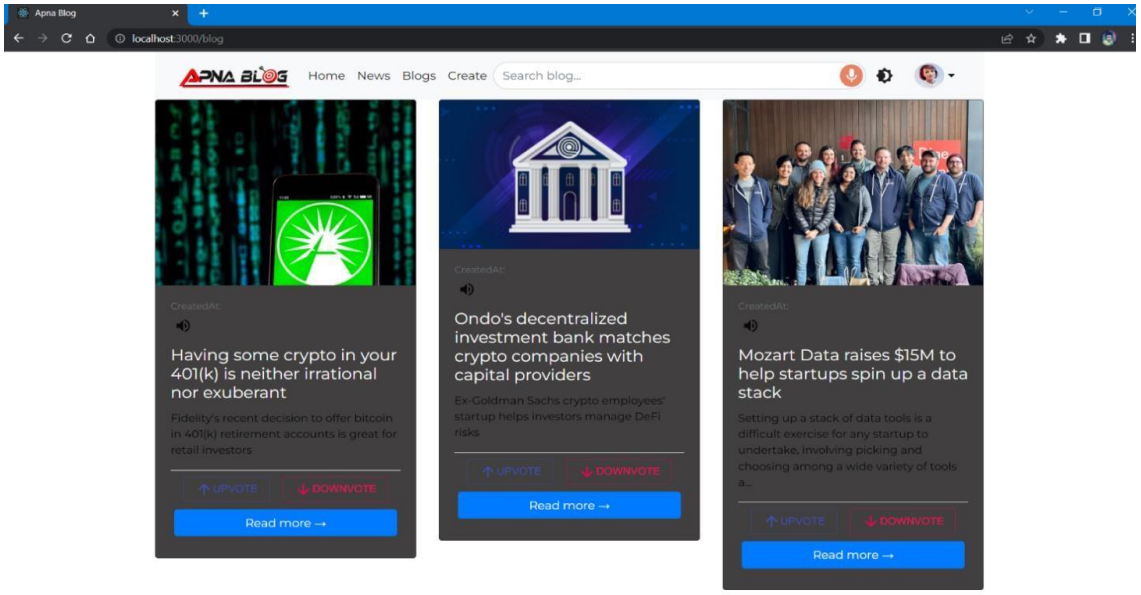
1) Click on Login/Register from nav-bar for sign-in or registering as a new user after successful login you will get landed on home.



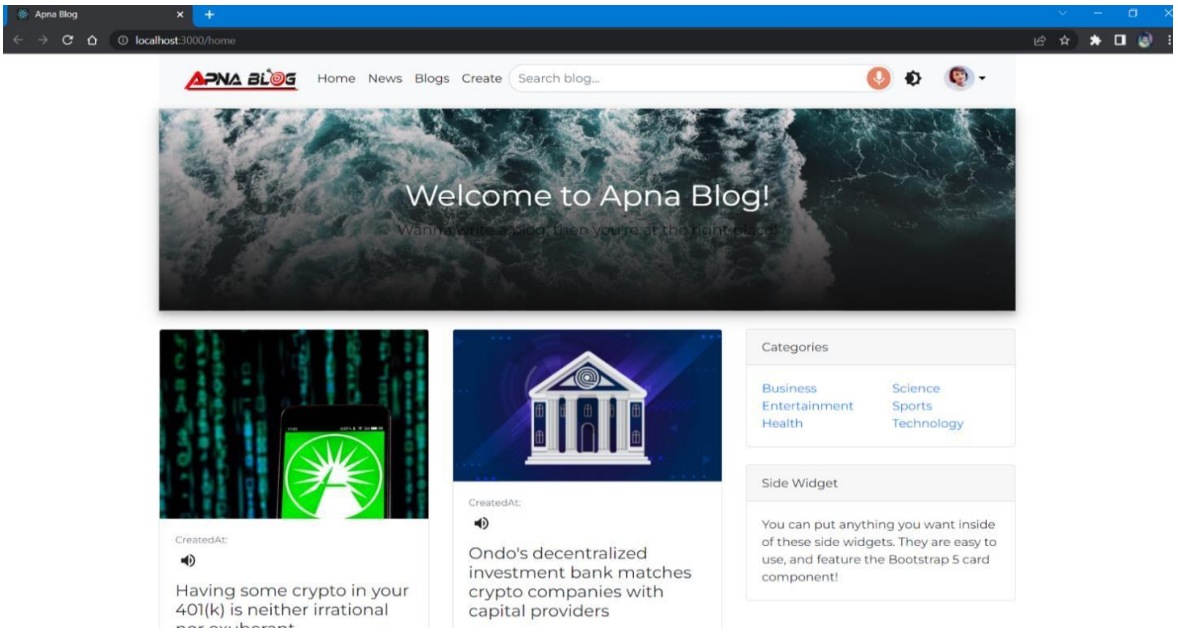


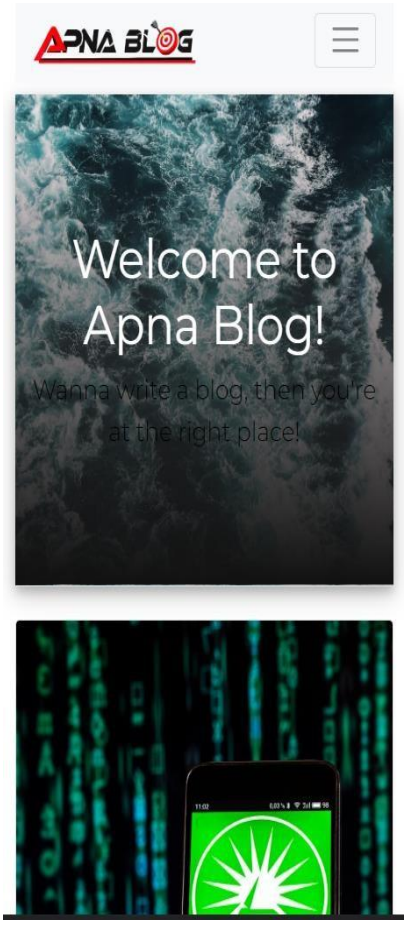
2) Click on Create bar for creating new blog.





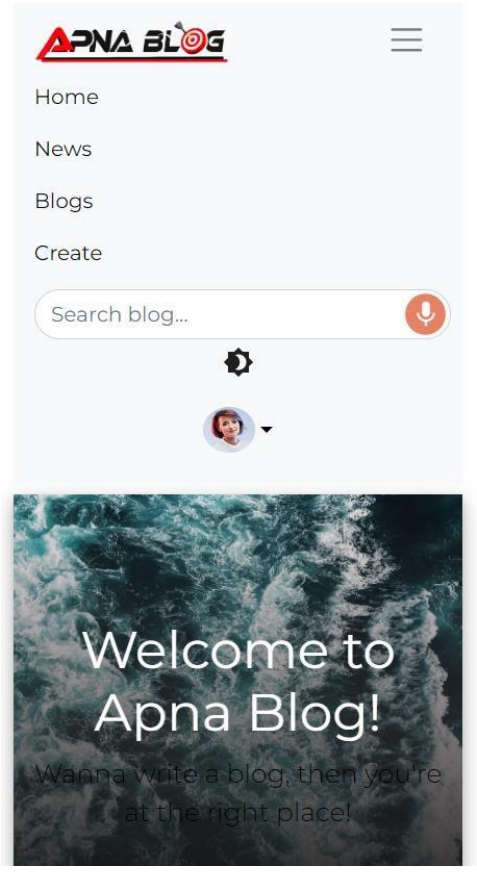
3) Click on icon to enable text to speech.



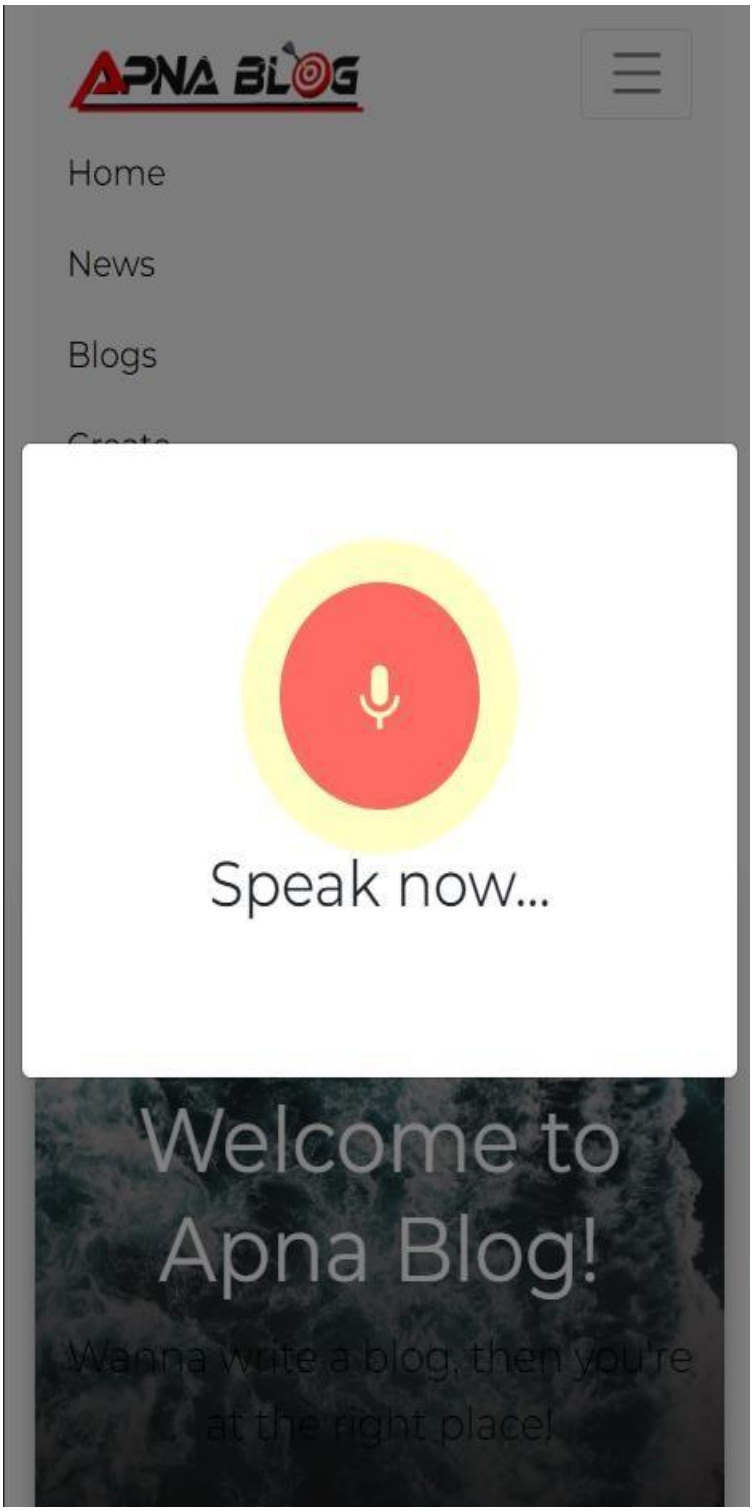


4) Click on home screen icon to land on homepage.

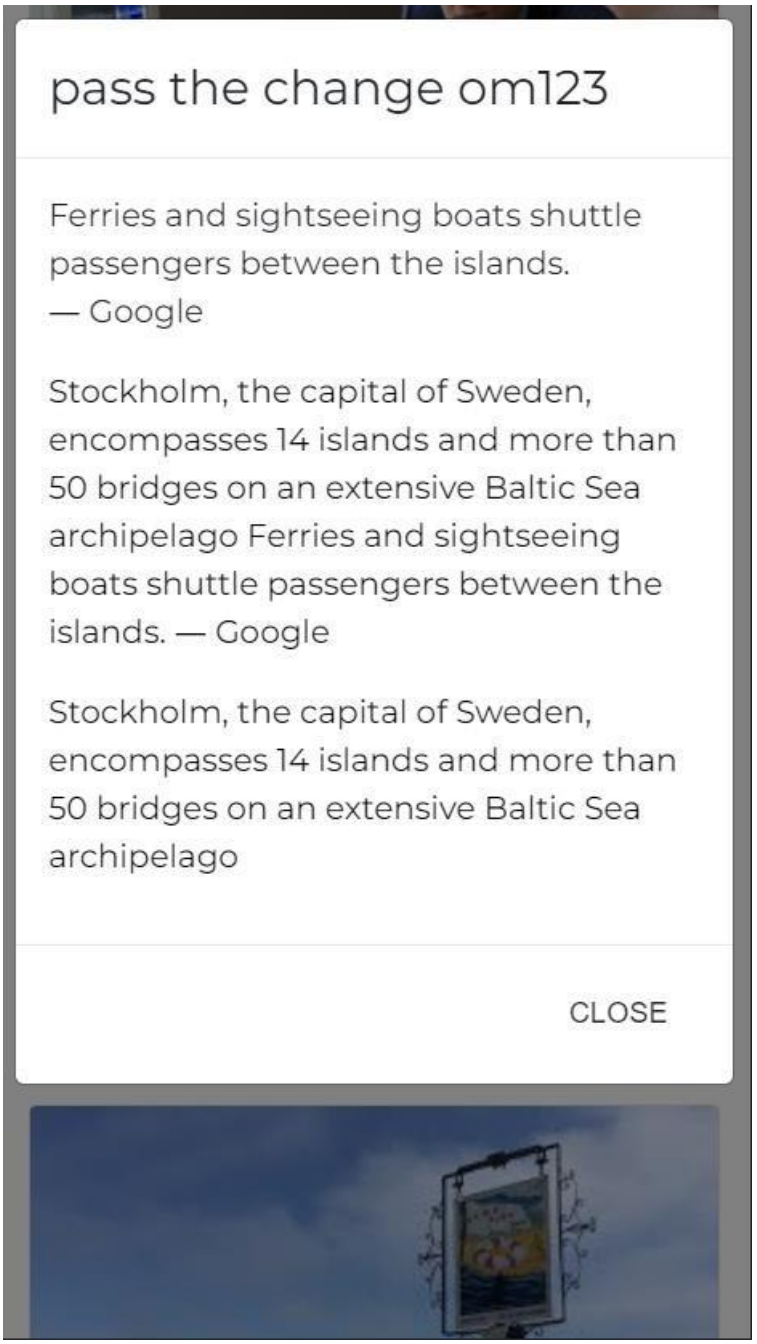
5) Click on upvote/downvote for voting a particular blog



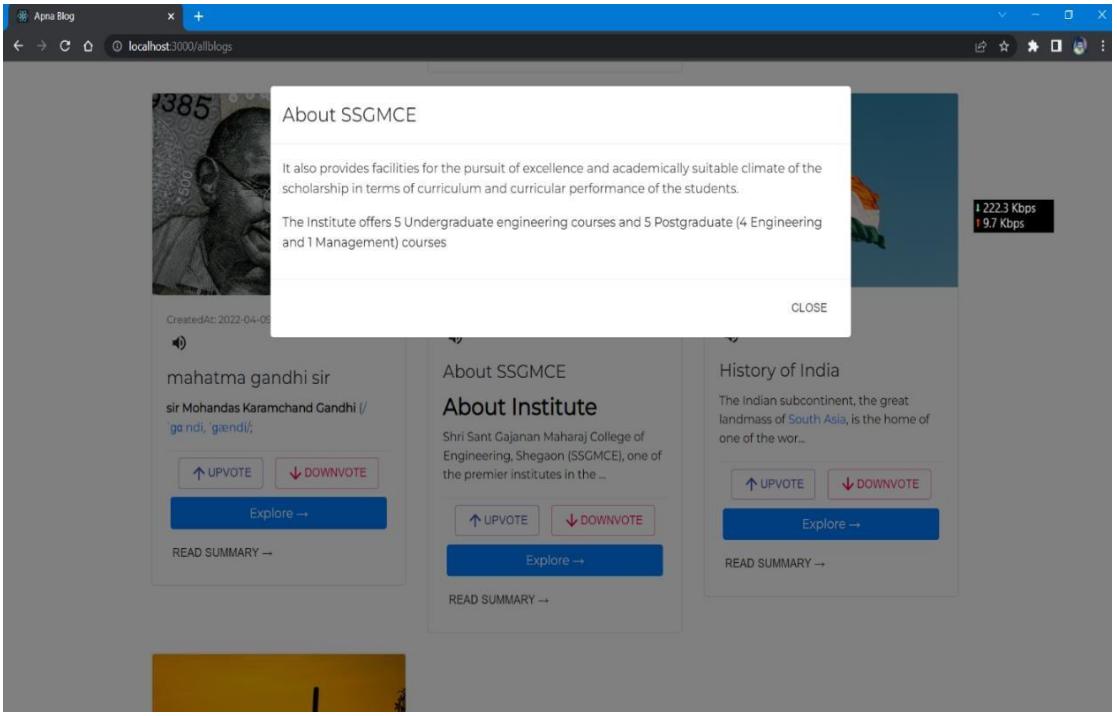
6) Responsive application view in Mobile device.

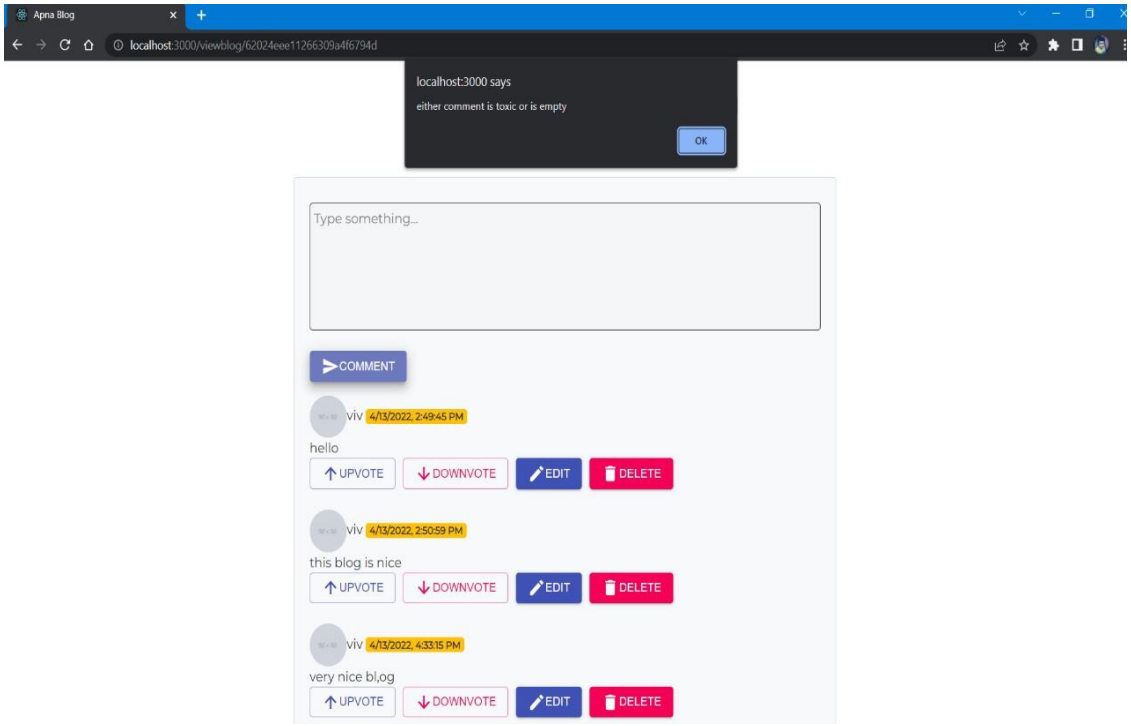


7) Click on mic icon for speech to text search of blog.



8) Click on Summary to display summary of a blog.





9) Type any comment in comment box and click on comment icon to submit it,If comment is offensive/toxic or empty message will be flashed “Either comment is toxic or is empty”

10) Click on edit/delete icon to edit or delete the comment

REFERENCES

[1] Classification of Online Toxic Comments Using Machine Learning Algorithms Published in: IEEE 2020 4th International Conference on Intelligent Computing and Control Systems.

[2] Extractive Text Summarization using sentence ranking IEEE Published in: 2019 International Conference on Data Science.

[3] Goularte, F. B., Nassar, S. M., Fileto, R., and Saggion, H. (2019). A text summarization method based on fuzzy rules and applicable to automated assessment. Expert Systems with Applications, 115:264–275.

[4] Mehta, P. and Majumder, P. (2018). Effective aggregation of various summarization techniques. Information Processing & Management, 54(2):145–158.

[5] Sanchez-Gomez, J. M., Vega-Rodr´ıguez, M. A., and Perez, ´ C. J. (2018). Extractive multi-document text summarization using a multi-objective artificial bee colony optimization approach. Knowledge-Based Systems, 159:1–8

[6] Text Summarization using Sentence Scoring Method IRJET 2017 International Research Journal of Engineering and Technology

[7] H. M. Saleem, K. P. Dillon, S. Benesch, and D. Ruths, "A Web of Hate: Tackling Hateful Speech in Online Social Spaces," 2017.

[8] Abbasi-ghalehtaki, R., Khotanlou, H., and Esmaeilpour, M. (2016). Fuzzy evolutionary cellular learning automata model for text summarization. Swarm and Evolutionary Computation, 30:11–26.

[9] M. Duggan, "Online harassment 2017," Pew Res., pp. 1–85, 2017, doi: 202.419.4372.

[10] Foul Language Comment Classification May- 2016 International Journal of Computational Engineering Research (IJCER).

[11] N. Moratanch, Dr. S. Chitrakala, "A Surveyon Abstractive Text Summarization." International Conference on Circuit, Powe and Computing Technologies (ICCPCT), 2016.

[12] Multi-Document Abstractive Summarization Based on Ontology IEEE 2015.

[13] Amancio, D. R. (2015). Probing the topological properties of complex networks modeling short written texts. PloS one, 10(2):e0118394

[14] Glavas, G. and ˇ Snajder, J. (2014). Event graphs for information retrieval and multi-document summarization. Expert systems with applications, 41(15):6904–6916

[15] Christensen, J., Soderland, S., Etzioni, O., et al. (2013). Towards coherent multi-document summarization. In Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: Human language technologies, pages 1163–1173.

[16] Alguliev, R. M., Aliguliyev, R. M., and Isazade, N. R. (2013). Multiple documents summarization based on evolutionary optimization algorithm. Expert Systems with Applications, 40(5):1675–1689.

[17] Wan, X. (2010). Towards a unified approach to simultaneous single-document and multi-document summarizations. In Proceedings of the 23rd international conference on computational linguistics, pages 1137–1145. Association for Computational Linguistics.

[18] Kavita Ganesan, ChengXiangZhai, Jiawei Han, "Opinosis: A Graph-Based Approach to Abstractive Summarization of Highly Redundant Opinions." Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 340- 348, 2010.

[19] Vishal Gupta, G.s. Lehal. "A survey of text mining techniques and applications", Journal of Emerging Technologies in Web intelligence,VOL 1,NO 1,6076,August 2009.

[20] Goldstein, J., Mittal, V., Carbonell, J., and Kantrowitz, M. (2000). Multi-document summarization by sentence extraction. In Proceedings of the 2000 NAACL-ANLP Workshop on Automatic summarization, pages 40–48. Association for Computational Linguistics.

[21] Goldstein, J. and Carbonell, J. (1998). Summarization:(1) using mmr for diversity-based reranking and (2) evaluating summaries. In Proceedings of a workshop on held at Baltimore, Maryland: October 13-15, 1998, pages 181–195.

[22] Baxendale,P.(1958). "Machine-made index for technical literature" –an experiment. IBM Journal of Research developement354-361

DISSEMINATION OF WORK

- A paper on Extractive Text Summarization in International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
  DOI: 10.48175/IJARSCT-3022

SOURCE CODE LISTINGS

The following are the folder structure that are used for blog application:

Client
- Node modules
- Public
- Src
  - Assets
  - Components
  - Helper
  - Hooks
  - Pages
  - Screens

Server

1. Controller: Controller includes core method of the functionality.

2. Repository: Repository includes the http request and response services.

3. Model: Model store the entity key and type of entity response.

4. Helpers: Helpers include some commonly used value or config element.

5. Elements: Elements includes the widget code of the element.

6. Pages: Used to write code for different pages.

Toxicity Code

```
import * as toxicity from '@tensorflow-models/toxicity';
import '@tensorflow/tfjs-backend-cpu';
import { useEffect, useState } from 'react';

/**
 * @returns Promise that resolves to return Toxicity ML Model
 **/

export const useToxicityClassifier = () => {
  const [toxicityModel, setToxicityModel] = useState();
  const [modelReady, setModelReady] = useState(false);
```

```
 useEffect(() => {
   const loadModel = async () => {
     const model = await toxicity.load();
     setToxicityModel(model);
     setModelReady(true);
   };
   loadModel();
 }, [modelReady]);

 return modelReady ? toxicityModel : null;
};
```

Summarizer code

```
import {stopWords, specialWords} from './stopWords.js';

const charCount = (word, char) => {
   let count = 0;
   for (let i = 0; i < word.length; i++) {
               if (word.charAt(i) === char) {
                           count++;
               }
   }
   return count;
}

const summarize = (textIn) => {
 let contents = textIn.split(' ');  //splitting of words
 //let      indivSentences      =      textIn.match(/[A-Z''][A-Za-z0-9\s''';:,%&@#$^*(){}[\]-
]*([.?!]+)|([.!?]+'')/g);
 // \s+[A-Z][A-Za-z0-9\s''';:,]*[.?!]
 // \s+[A-Za-z,;'\''\s]+[.?!]
 let indivSentences = textIn.split(/[.!?]+''?(?=\s[A-Z])/);
```

```
console.log("indivSentences",indivSentences); //splitting of sentences

const wordFreq = new Map();

for (let i = 0; i < contents.length; i++) {
 let word = contents[i];
 word = word.toLowerCase();

 //if word has 1 period and isn't in list of known words with .
 //what this does is make sure we count words at end of sentence
 if (charCount(word, '.') === 1 && word.endsWith('.') && !specialWords.includes(word))
{
        word = word.substring(0, word.length-1);
 } else if (word.endsWith(';') || word.endsWith(',') || word.endsWith(':')) {
        word = word.substring(0, word.length-1);
 }

 //no word should have unnecessary trailing '.' after this point
 if (!wordFreq.has(word) && !stopWords.includes(word)) {
        wordFreq.set(word, 1);
 } else if (wordFreq.has(word)) {
        let updatedVal = wordFreq.get(word) + 1;
    wordFreq.set(word, updatedVal);
 }
}

console.log("wordFreq",wordFreq);
//sortedMap stores the frequency of each word in descending order
let sortedMap = new Map([...wordFreq].sort(([k, v],[k2, v2]) => {
return (v > v2) ? -1 : ((v < v2) ? 1 : 0);
}));
console.log("sortedMap",sortedMap);

let scoreMap = new Map();
```

```
//loop through sentences and calculating score of each sentence
for (let i = 0; i < indivSentences.length; i++) {
 let score = 0;
 let subWords = indivSentences[i].split(' ');
 for (let j = 0; j < subWords.length; j++) {
            if (sortedMap.get(subWords[j]) !== undefined) {
                    score += sortedMap.get(subWords[j]);
            }
 }
 scoreMap.set(indivSentences[i], score);
}

let sortedScoreMap = new Map([...scoreMap].sort(([k, v],[k2, v2]) => {
 return (v > v2) ? -1 : ((v < v2) ? 1 : 0);
}));

//let topScores = Array.from(sortedScoreMap.values()).slice(0,5); //threshold
//let topScores = Array.from(sortedScoreMap.values()).slice(0,2); //threshold
let topScores = Array.from(sortedScoreMap.values()).slice(0,1); //threshold

console.log("scoreMap",scoreMap);
console.log("sortedScoreMap",sortedScoreMap);
console.log("topScores",topScores);
let output = '';
let temp = [];
for (let i = 0; i < indivSentences.length; i++) {
  if ( topScores.includes(scoreMap.get(indivSentences[i])) ) {
    temp.push(indivSentences[i]);
    output += indivSentences[i];
  }
}
console.log(temp);
```

```
 return output;
}

export default summarize;


//wordFreq (map) stores the frequency of each word
//sortedMap stores the frequency of each indivisual word in the document in descending order

//scoreMap and sortedScoreMap are for sentence scores
//indivisualSentence array stores all sentences in document

//topscores stores the top 5 sentence's scores
```

LIST OF STUDENTS

	Name: Vivek Bhore Mail:vivekbhore50gmail.com Mobile No :9370117371 Address : Dhamangaon Badhe, Buldana Maharashtra
	Name: Rutik Gawande Email: mirutikgawande@gmail.com Mobile No: 7038903803 Address: Hinganghat, Maharashtra
	Name: Pratik Bondre Email: pratikbondre@gmail.com Mobile No: 9754763887 Address: Nagpur Maharashtra
	Name: Vrushabh Guntiwar Email:vrushabhguntiwar@gmai.com Mobile No: 8770540346 Address: Chandrapur , Maharashtra