

Software Engineering Department

Ort Braude College

Capstone Project Phase B – 61999

**ShopLINK**

**Authors:**

Julia Shbat – Julia.Shbat@e.braude.ac.il

Daniel Vardimon– Daniel.Vardimon@e.braude.ac.il

**Supervisor:**

Dr. Zakharia Frenkel

**24-2-D-27**

**Table of Contents**

[1. INTRODUCTION 5](#_Toc124978537)

[2. RELATED WORKS 6](#_Toc124978540)

[2.1. Zoko 6](#_Toc124978541)

[2.2. Zixflow 6](#_Toc124978542)

[2.3. Interakt 6](#_Toc124978543)

[3. THE NETWORK 7](#_Toc124978544)

[3.1. Natural Language Processing (NLP) 7](#_Toc124978545)

[3.2. Components of the NLP Algorithm 7](#_Toc124978546)

[3.3. NLP Techniques & Algorithms 8](#_Toc124978547)

[4. TECHNOLOGY REVIEW 9](#_Toc124978548)

[Table 4.1. Google Cloud NLP 9](#_Toc124978545)

[5. TECHNOLOGY CHOICES 9](#_Toc124978548)

[5.1 BACK-END NODEJS 9](#_Toc124978549)

[5.1.1. ADVANTAGES OF NODE.JS 9](#_Toc124978550)

[5.1.2. DISADVANTAGES OF NODE.JS 10](#_Toc124978551)

[5.2. BACK-END MONGODB 10](#_Toc124978552)

[5.2.1 ADVANTAGES OF MONGODB 11](#_Toc124978553)

[5.2.2 DISADVANTAGES OF MONGODB 11](#_Toc124978553)

[5.3. FRONT-END REACT-JS 11](#_Toc124978554)

[5.3.1 ADVANTAGES OF REACT-JS 11](#_Toc124978555)

[5.3.2 DISADVANTAGES OF REACT-JS 11](#_Toc124978556)

[5.4.2 ADVANTAGES OF VISUAL STUDIO CODE 12](#_Toc124978555)

[5.4.3 DISADVANTAGES OF VISUAL STUDIO CODE 12](#_Toc124978556)

[5.5 EXPRESS-JS 13](#_Toc124978557)

[5.6 Socket.io 13](#_Toc124978557)

[6. Engineering Process 13](#_Toc124978561)

[6.1. QR Code Authentication 13](#_Toc124978562)

[6.2. Persistent WhatsApp Connection 14](#_Toc124978563)

[6.3. Receiving Incoming Messages 14](#_Toc124978564)

[6.4. Message Classification using Google Cloud NLP 1](#_Toc124978565)4

[6.5. Customer Search & Selection 1](#_Toc124978565)4

[6.6. Editing & Updating Customer Records 1](#_Toc124978565)4

[6.7. Deleting Customers 1](#_Toc124978565)4

[6.8. Broadcast Messaging System 1](#_Toc124978565)5

[6.7. Real-Time WebSocket Updates 1](#_Toc124978565)5

[6.7. Handling Disconnects & Recovery and Safe Logout & Restart 1](#_Toc124978565)5

[7. CHALLENGES 16](#_Toc124978566)

[7.1. WhatsApp API Limitations 16](#_Toc124978567)

[7.2. Time Management and Team Coordination 16](#_Toc124978567)

[7.3. Data Availability and Collection 16](#_Toc124978567)

[8. PROJECT STRUCTURE 17](#_Toc124978569)

[8.1 MERN Stack 17](#_Toc124978570)

[9. SOFTWARE ENGINEERING DOCUMENTS 18](#_Toc124978583)

[9.1 DIAGRAMS 18](#_Toc124978584)

[9.1.1 Package diagram 18](#_Toc124978585)

[9.1.2 FlowChart diagram 18](#_Toc124978586)

[10. TEST PLANS 20](#_Toc124978625)

[10.1. LOGIN TESTS 20](#_Toc124978626)

[10.2. MANAGER TESTS 21](#_Toc124978627)

[10.3. BROADCAST MESSAGE TESTS 21](#_Toc124978627)

[10.4. SALES & ANALYTICS TESTS 22](#_Toc124978627)

[10.5. WHATSAPP WEB INTEGRATION TESTS 23](#_Toc124978627)

[11. Conclusion 24](#_Toc124978630)

[12. User Documentation 25](#_Toc124978630)

[12.1 User’s guide - Operating instruction 25](#_Toc124978584)

[12.1.1 General description 25](#_Toc124978585)

[12.1.2 MANAGER SCREENS 26](#_Toc124978585)

[13. Maintenance Guide 28](#_Toc124978630)

[14. REFERENCES 29](#_Toc124978630)

**Abstract.** *With recent increases in online purchases, many businesses allow customers to contact them through WhatsApp for support and marketing. This creates a big burden on the business and may overload the business' WhatsApp, making it difficult to keep up with the customers. The business owners can have difficulties seeing the customers messages on time, and this may cause problems with communication and availability for the customers, and may compromise the service quality of the business. Along with these difficulties, there is a need for methodically sending the correct advertisements to the corresponding customers according to their interests. There are some solutions that exist, but those don't exactly solve all of the problems, especially they are not able to automatically select groups of users, for example, for sending some marketing information.*

*In our project we are going to develop a new product with a standard set of functionalities as well as involving the clustering and the topic extraction algorithms for proceeding of the chats with users for creating different groups of users.*

*The standard functionalities include a comprehensive dashboard for managing customer interactions, detailed analytics to monitor engagement, and tools for organizing customer conversations using labels and tags.*

**1. INTRODUCTION**

With the surge in online shopping, businesses increasingly use platforms like WhatsApp for customer interactions, leveraging its convenience and popularity. This shift, however, presents significant challenges. The sheer volume of messages can overwhelm businesses, causing delays in response times and leading to potential dissatisfaction among customers. Effective communication is crucial for maintaining service quality, and any delay can negatively impact the customer experience.

Additionally, businesses face the challenge of targeting the right customers with appropriate advertisements. As customer bases grow, it becomes increasingly difficult to manually manage and personalize marketing efforts. There is a need for an efficient, automated solution that not only manages customer communications but also intelligently clusters customers based on their interests for targeted marketing.

Currently, various tools and platforms provide basic solutions for managing customer interactions and marketing via WhatsApp. These tools often include simple automated responses and basic customer segmentation features. However, they typically lack advanced capabilities for grouping customers based on detailed interest analysis. Manual processes are still prevalent, which can lead to inefficiencies and missed opportunities in personalizing customer engagement.

Our proposed application addresses these limitations by integrating advanced communication methods and data analysis techniques. The system gathers data on customer interests, which is stored in a database and analyzed using keywords and deep learning algorithms. These algorithms cluster customers into groups based on shared interests, enabling highly personalized marketing efforts.

By automating the clustering process and tailoring advertisements to specific customer groups, businesses can improve response times and enhance the relevance of their marketing messages. This leads to a more streamlined communication process and boosts overall customer satisfaction.

The primary stakeholders in this solution are business owners and their customers. Business owners gain a powerful tool to manage customer interactions more efficiently, reducing the manual workload and improving marketing precision. This not only enhances operational efficiency but also positively impacts the business's bottom line by increasing customer engagement and retention.

Customers, on the other hand, benefit from more timely responses and receive advertisements tailored to their specific interests, leading to a more satisfying and personalized shopping experience. This improved interaction fosters customer loyalty and strengthens the relationship between businesses and their clientele.

**2. RELATED WORKS**

The WhatsApp Business API enables businesses to send bulk messages and connect multiple devices to a single WhatsApp Business account, a feature not available with the standard WhatsApp Business App. This capability is essential, as WhatsApp statistics indicate that approximately 175 million users contact companies via WhatsApp daily.

**2.1. Zoko**

Zoko is a marketing application that allows selling products over WhatsApp. It offers features such as broadcasting messages, automated responses, and sales tracking, which help businesses streamline their communication and marketing efforts.The application also offers tools for making a product catalog on WhatsApp, a chat screen and an AI chatbot for communicating with customers, as well as tools for analytics to help sellers and business owners track and monitor their sales. Zoko includes integration with a few other tools for marketing such as FlowHippo and Shopify, adding the benefits of such tools to the app.

**2.2. Zixflow**

Zixflow is a powerful platform designed to streamline document review and approval processes for businesses. It offers features such as automated workflows, real-time collaboration, and detailed tracking of document statuses, which enhance efficiency and reduce manual tasks. Zixflow's robust security measures ensure that sensitive documents are protected, and its comprehensive reporting tools provide valuable insights into workflow performance. It allows users to segment their audience, add tags and create custom contact lists for WhatsApp broadcasts, and offers a centralized inbox and automated messaging, helping the users with customer support.

**2.3. Interakt**

Interakt is a versatile app designed to enhance customer engagement through seamless integration with WhatsApp. It offers features such as automated workflows and real-time messaging, which streamline communication and improve efficiency. Interakt's advanced security protocols ensure that customer data is protected, and its robust analytics tools provide valuable insights into user interactions. Additionally, it allows businesses to set up WhatsApp chatbots with Chat GPT with automated replies and automated tasks, run WhatsApp ads and AI generated ads with auto target, and offers CRM and shared inbox for improved customer support. Interakt integrates with many platforms including Shopify, Google Sheets and more.

**3. THE NETWORK**

**3.1** **Natural Language Processing (NLP)**

Natural Language Processing (NLP) is a field of artificial intelligence (AI) that enables computers to understand, interpret, and generate human language. NLP combines computational linguistics with machine learning, deep learning, and statistical models to process and analyze text and speech data. The goal of NLP is to bridge the gap between human communication and machine understanding, allowing computers to interact with text or spoken language in a meaningful way.

**3.2** **Components of the NLP Algorithm**

The NLP algorithm consists of several key components, each playing a crucial role in text processing:

1. Text Preprocessing:  
   Before applying advanced NLP techniques, raw text data undergoes preprocessing to ensure accuracy and efficiency. This includes:
   * Tokenization: Splitting text into individual words or sentences.
   * Stopword Removal: Eliminating common words (e.g., "the," "is," "and") that do not contribute much meaning.
   * Lemmatization/Stemming: Converting words to their root form (e.g., "running" → "run").
   * Lowercasing: Converting all text to lowercase to maintain consistency.
   * Punctuation and Special Character Removal: Cleaning text by removing unnecessary symbols.
2. Feature Extraction:  
   After preprocessing, the text is converted into a numerical format that a machine learning model can process. Common techniques include:
   * Bag of Words (BoW): Representing text as word frequency counts.
   * TF-IDF (Term Frequency-Inverse Document Frequency): Assigning weights to words based on their importance in a document.
   * Word Embeddings (e.g., Word2Vec, GloVe): Mapping words into high-dimensional vectors based on context and meaning.
3. Text Classification & Understanding:  
   The algorithm applies classification techniques to categorize text into predefined labels or extract relevant information. Methods include:
   * Rule-Based Approaches: Using predefined linguistic rules to analyze text.
   * Machine Learning Models: Employing algorithms like Naïve Bayes, Support Vector Machines (SVM), or Decision Trees for classification.
   * Deep Learning Models: Utilizing neural networks such as Recurrent Neural Networks (RNNs) and Transformers (e.g., BERT, GPT) for advanced text understanding.
4. Named Entity Recognition (NER):  
   This technique identifies and classifies key entities in text, such as names, dates, locations, and organizations. NER improves the system’s ability to extract meaningful data from unstructured text.
5. Sentiment Analysis:  
   The NLP algorithm determines the sentiment expressed in a given text (e.g., positive, negative, or neutral). This is useful for analyzing customer feedback, reviews, and social media content.
6. Text Generation:  
   In certain cases, the NLP model can generate human-like responses or summarize large bodies of text using deep learning techniques such as Transformer-based models (e.g., GPT, T5).
7. Speech Processing (Speech-to-Text & Text-to-Speech) NLP is also used in speech recognition and synthesis:

* Speech-to-Text (STT): Converts spoken language into written text (e.g., voice assistants like Siri and Google Assistant).
* Text-to-Speech (TTS): Converts written text into spoken language using neural synthesis.

**3.3** **NLP Techniques & Algorithms**

Various algorithms are used in NLP to enhance understanding and processing of text. Some widely used approaches include:

* Naïve Bayes Classifier: A probabilistic model for text classification.
* Support Vector Machines (SVM): A machine learning model for categorizing text.
* Recurrent Neural Networks (RNNs) & Long Short-Term Memory (LSTM): Deep learning models for sequence-based tasks.
* Transformers (e.g., BERT, GPT-4): Advanced models that provide contextual understanding of text.

**4. TECHNOLOGY REVIEW**

**4.1. Google Cloud NLP**

API Reference: https://cloud.google.com/natural-language/docs

Price: $300 in free credits to try Google Cloud products

The Cloud Natural Language API provides natural language understanding technologies to developers, including sentiment analysis, entity analysis, entity sentiment analysis, content classification, and syntax analysis. This API is part of the larger Cloud Machine Learning API family.

**5. TECHNOLOGY CHOICES**

**5.1 BACK-END NODEJS**

Node.js is an open-source, cross-platform environment that allows JavaScript to run server-side. It's a popular choice for a wide range of projects! Node.js runs outside of the browser using the V8 JavaScript engine, which powers Google Chrome. This gives Node.js a significant performance edge.

A Node.js application runs in a single process without creating a new thread for each request. Most Node.js libraries are built using non-blocking patterns, and its standard library includes asynchronous I/O primitives that prevent JavaScript code from blocking. This means Node.js doesn't wait for I/O operations, like reading from the network or a database, to complete before continuing its execution. Instead, it resumes once the response is received.

This approach simplifies the complexity of managing thread concurrency, which can be a major source of bugs, allowing Node.js to efficiently handle thousands of simultaneous connections with just one server. Additionally, Node.js is advantageous for the millions of frontend developers who are familiar with JavaScript, enabling them to write both server-side and client-side code without having to learn a new language.

Now, let's examine the key advantages and disadvantages of using Node.js for server-side development.

#### 5.1.1. ADVANTAGES OF NODE.JS

1. High performance for real-time applications.
2. Scalable solutions for modern applications.
3. Cost-effective with full-stack JavaScript.
4. Strong community support to streamline development.
5. Simple to learn and adapt quickly.
6. Facilitates the creation of cross-functional teams.
7. Enhances application response time and overall performance.
8. Reduces time-to-market for your applications.
9. Extensibility to meet custom requirements.
10. Quick caching reduces loading times.
11. Supports the development of cross-platform applications.

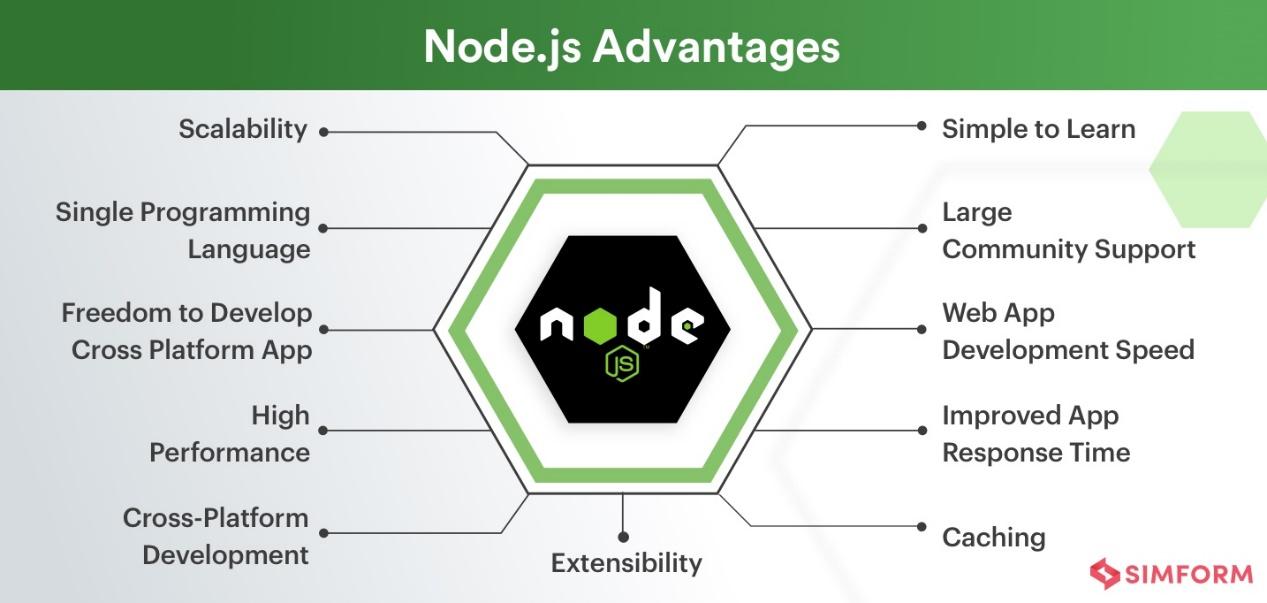


Figure 1 Node.js Advantages*, Reference 1*

#### 

#### 5.1.2. DISADVANTAGES OF NODE.JS

1. Reduced performance when handling heavy computational tasks.
2. Unstable API can lead to frequent code changes.
3. Asynchronous programming in Node.js can make code maintenance challenging.
4. Insufficient library support can pose risks to your code—choose wisely.
5. High demand for Node.js developers with relatively few experienced professionals.

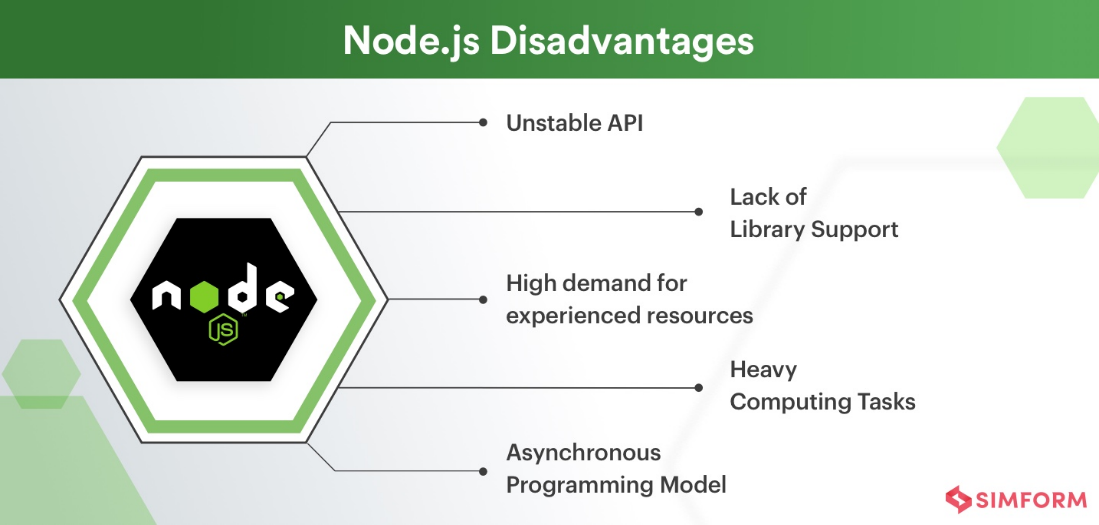


Figure 2 Node.js Disadvantages*, Reference 1*

תמונה שמכילה גופן, לוגו, גרפיקה, עיצוב

התיאור נוצר באופן אוטומטי

**5.2. BACK-END MONGODB**

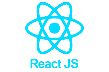
MongoDB is an open-source, cross-platform, document-oriented database. As a NoSQL database, MongoDB stores data in JSON-like documents, which may or may not adhere to a predefined schema. The database is developed by MongoDB Inc. under the Server-Side Public License (SSPL), which some distributions consider non-free.

**5.2.1 ADVANTAGES OF MONGODB**

1. Schema-less − MongoDB is a document database where a single collection can contain various documents, each with different fields, content, and sizes.
2. Clear structure of individual objects.
3. No need for complex joins.
4. Powerful querying capabilities. MongoDB supports dynamic queries on documents through a document-based query language that is nearly as robust as SQL.

#### 5.2.2 DISADVANTAGES OF MONGODB

1. Limited support for complex transactions − MongoDB doesn't naturally support multi-document transactions, which can be a limitation for applications requiring complex operations across multiple documents.
2. High memory usage − MongoDB's approach to data storage and indexing can lead to higher memory consumption compared to relational databases.

**5.3. FRONT-END REACT-JS **

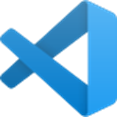
React.js is a popular JavaScript library used for building interactive and dynamic user interfaces, particularly single-page applications. It focuses on creating reusable UI components, which make development more efficient and maintainable. React's virtual DOM ensures high performance by minimizing direct interactions with the real DOM, leading to faster updates and rendering. It also enables seamless integration with other libraries or frameworks, making it highly versatile for front-end development.

**5.3.1 ADVANTAGES OF REACT-JS**

1. Component-Based Architecture: React promotes reusability by breaking the UI into modular components, which simplifies maintenance and scalability.
2. Virtual DOM: React’s virtual DOM improves performance by efficiently updating and rendering only the components that change, rather than the entire page.
3. Strong Community and Ecosystem: With a large developer community, extensive documentation, and plenty of third-party libraries, React has strong support for building complex applications.

**5.3.2 DISADVANTAGES OF REACT-JS**

1. Learning Curve: React has a steeper learning curve, especially for new developers, due to the concepts like JSX, component lifecycle, and state management.
2. Overhead for Small Applications: For small, simple projects, React can be overkill, as its advanced features and architecture might not be fully utilized.
3. State Management Complexity: As applications grow in size, managing the state across multiple components can become complex, often necessitating additional libraries like Redux or Context API for effective state management.

**5.4 VISUAL STUDIO CODE FRAMEWORK**

Visual Studio Code (VS Code) is a source-code editor developed by Microsoft, built on the Electron framework, and available for Windows, Linux, and macOS. It offers a range of features such as debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and integrated Git support.

**5.4.1 FEATURES**

Visual Studio Code is a versatile source-code editor compatible with various programming languages, including C#, Java, JavaScript, Go, Node.js, Python, C++, C, Rust, and Fortran. It is built on the Electron framework, which is commonly used for developing Node.js web applications running on the Blink layout engine. The editor component, codenamed "Monaco," is the same as the one used in Azure DevOps (formerly Visual Studio Online and Visual Studio Team Services).

Out of the box, Visual Studio Code provides basic support for most popular programming languages. This support includes features like syntax highlighting, bracket matching, code folding, and customizable snippets. Additionally, VS Code comes with IntelliSense for JavaScript, TypeScript, JSON, CSS, and HTML, and offers debugging capabilities for Node.js. Developers can further extend its functionality by installing extensions available on the VS Code Marketplace, which add support for additional languages and tools.

#### 5.4.2 ADVANTAGES OF VISUAL STUDIO CODE

1. Cross-platform availability − VS Code runs on Windows, Linux, and macOS, providing a consistent development environment across different operating systems.
2. Extensibility − The rich ecosystem of extensions available on the VS Code Marketplace allows developers to customize and enhance their coding environment to fit their specific needs.
3. Integrated Git support − VS Code includes built-in Git support, allowing for seamless version control directly within the editor.
4. IntelliSense − Provides intelligent code completion, parameter info, and quick info for supported languages, improving developer productivity.
5. Lightweight and fast − Despite its powerful features, VS Code is known for being lightweight and responsive, making it suitable for both small and large projects.
6. Active community and frequent updates − With an active community and regular updates from Microsoft, VS Code continuously evolves with new features and improvements.

#### 5.4.3 DISADVANTAGES OF VISUAL STUDIO CODE

1. Resource consumption − As an Electron-based application, VS Code can be more resource-intensive compared to native editors, which may affect performance on lower-end systems.
2. Limited built-in features − While VS Code is feature-rich, some advanced features require additional extensions, which may not always integrate seamlessly or perform optimally.

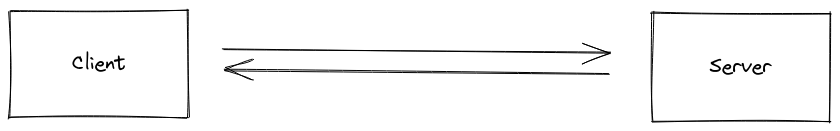
**5.5 EXPRESS-JS**

תמונה שמכילה גופן, לוגו, גרפיקה, סמל

התיאור נוצר באופן אוטומטיExpress.js is a lightweight and flexible web application framework for Node.js that simplifies the process of building web servers and APIs. It provides a set of robust features for handling HTTP requests, middleware, routing, and creating RESTful services. Express is unopinionated, meaning it doesn’t enforce any specific project structure, giving developers the freedom to organize code as they see fit. Its simplicity, combined with the vast ecosystem of Node.js libraries, makes Express a popular choice for building scalable, high-performance web applications.

**5.6 Socket.io**

Socket.io is a powerful JavaScript library that enables real-time, bidirectional communication between web clients and servers. Built on top of WebSockets, it provides a simple API for handling events, message broadcasting, and automatic reconnections. Socket.io abstracts the complexities of WebSockets while offering fallback mechanisms for older browsers that do not support them. It is commonly used for chat applications, live notifications, collaborative tools, and real-time data streaming. With built-in support for namespaces and rooms, Socket.io allows efficient management of multiple connections, making it a preferred choice for building interactive and dynamic web applications.



**6. Engineering Process**

Our development process began with an in-depth study of deep neural network structures and the frameworks, which we intended to utilize for algorithm development.

The system is designed to integrate WhatsApp Web with Google Cloud NLP, enabling real-time message classification, structured data storage, and WhatsApp-based communication. The architecture consists of a backend server (Node.js, Express.js, MongoDB), real-time messaging (Socket.io, WhatsApp Web API), and a frontend interface.

**6.1. QR Code Authentication**

* When a user logs in, the system generates a QR code that must be scanned via WhatsApp Web.
* After scanning, WhatsApp Web establishes a persistent connection.

### If the user logs out, a new QR code is generated for re-authentication.

### 6.2. Persistent WhatsApp Connection

### The system maintains an active session, ensuring continuous message reception.

### If WhatsApp Web disconnects, the system automatically attempts reconnection.

### The user can manually log out or restart the client as needed.

### To ensures a seamless and secure connection to WhatsApp Web and, prevents manual re-login every time the system is used.

### 6.3 Receiving Incoming Messages

### WhatsApp messages are captured in real time.

### Messages are transmitted via WebSocket (Socket.io) to update the frontend.

# 6.4. Message Classification using Google Cloud NLP

Each received message is processed using Google Cloud NLP. The system analyzes the message to extract:

### Intent classification (e.g., order inquiry, complaint, product request).

### Sentiment analysis (positive, neutral, or negative).

### Key entities (e.g., product names, locations, brands).

The classification results are stored in MongoDB for future reference.

Allows AI-driven message categorization for better support and, enables businesses to quickly understand customer intent.

# 6.5. Customer Search & Selection

### Users can search for customers by name or phone number.

### The system filters results dynamically, allowing for quick selection.

### Once a customer is selected, their details are displayed for editing.

# 6.6. Editing & Updating Customer Records

### Users can modify a customer’s name, phone number, and email.

### Changes are saved in the database for future use.

# 6.7. Deleting Customers

### Users can remove a customer from the system.

### The deleted customer’s conversation history remains stored for analytics.

To ensures accurate and up-to-date customer records and, allows businesses to track customer interactions efficiently.

# 6.8. Broadcast Messaging System

### Customers are categorized based on tags (e.g., promo, electronics, computers).

### Users can select tags, and the system automatically filters customers.

### Users can compose a custom message.

### The system prevents empty messages from being sent.

### Messages are sent to all selected customers.

### The system displays a confirmation alert with recipient details.

To ensures targeted and effective communication and, allows businesses to send promotions, notifications, and bulk updates.

# 6.9. Real-Time WebSocket Updates

WebSockets enable instant updates for:

### New WhatsApp messages.

### QR Code changes.

### WhatsApp Web connection status.

To ensures real-time synchronization between WhatsApp Web and the system and, improves user experience by providing instant updates.

# 6.10. Handling Disconnects & Recovery and Safe Logout & Restart

### If WhatsApp Web disconnects, the system automatically attempts reconnection.

### If the user logs out, a new QR code is generated for authentication.

### Users can log out manually via the system.

### If required, the system restarts the WhatsApp client safely.

Prevents session loss and manual re-login issues and, ensures a stable and secure WhatsApp Web integration.

**7. CHALLENGES**

Throughout the development of our project, we encountered multiple challenges. Below, we describe these challenges and the strategies we implemented to overcome them.

**7.1. WhatsApp API Limitations**

One of the primary challenges we faced was integrating WhatsApp into our system. Initially, we planned to use the official WhatsApp API to manage customer interactions and automate messages. However, we discovered that the API had significant limitations, including access restrictions and complex verification requirements. Even after reaching out to the official support team, we were unable to resolve these issues.

To work around this challenge, we decided to utilize WhatsApp Web instead. By leveraging browser automation techniques, we were able to simulate user interactions and extract the necessary data for processing. While this approach was not as seamless as an official API integration, it provided us with the functionality we needed to continue our project without major disruptions.

**7.2. Time Management and Team Coordination**

Another significant challenge was coordinating our team’s efforts. Each team member had different schedules, and we often struggled to find common time slots for collaboration. Additionally, due to the complexity of the project, there were instances where individual tasks took longer than expected, leading to delays in our overall progress.

To address this challenge, we adopted a structured approach to time management. We divided tasks among team members based on their availability and expertise, ensuring that everyone had a clear responsibility. Additionally, we designated one specific day each week for a team meeting, where we could synchronize our progress, discuss any blockers, and adjust our strategy if needed. Communication was maintained through Microsoft Teams and WhatsApp, allowing us to stay connected and resolve issues efficiently, even when we couldn't meet in person.

**7.3. Data Availability and Collection**

Another major obstacle was obtaining real-world data for testing our system. Since we did not have a Business WhatsApp account, we lacked access to actual customer conversations, which made it difficult to validate our clustering and topic extraction algorithms. Without a sufficient dataset, the accuracy of our system’s ability to group users and send targeted advertisements was uncertain.

To address this, we generated simulated chat data based on typical customer-business interactions. We analyzed publicly available datasets and business communication patterns to create realistic sample conversations. While this was not an exact replica of real business interactions, it allowed us to test and refine our clustering algorithms. Additionally, we designed our system to be flexible so that once real business data becomes available, we can easily integrate and test it without major modifications.

**8. PROJECT STRUCTURE**

As technology evolves rapidly, user expectations and standards are continuously shifting. To keep up, developers strive to create applications that are more responsive, interactive, and enjoyable. A key approach to achieving this is through the use of technology stacks—carefully selected sets of functions and technologies designed to work seamlessly together. By leveraging advanced technology stacks, developers can ensure their applications are reliable, efficient, and capable of meeting the high demands of modern users.

**8.1. MERN Stack**

The MERN stack is a powerful combination of four technologies: MongoDB, Express.js, React.js, and Node.js. This stack allows developers to build a full-stack application with a cohesive 3-tier architecture (frontend, backend, database) using only JavaScript and JSON. By utilizing the MERN stack, developers can create robust and efficient applications that cater to a variety of needs and handle complex tasks effectively.

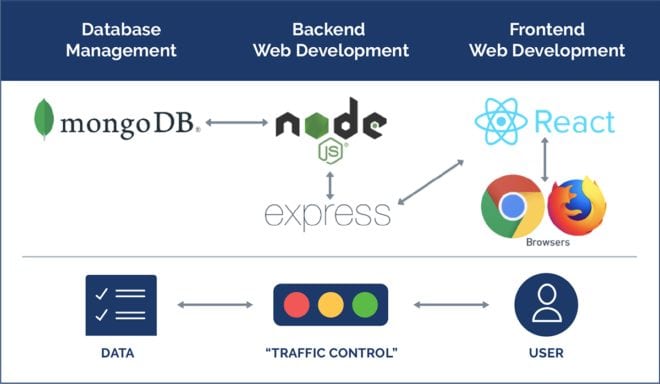


Figure 3 How does MERN work*, Reference 1**0*

**9. SOFTWARE ENGINEERING DOCUMENTS**

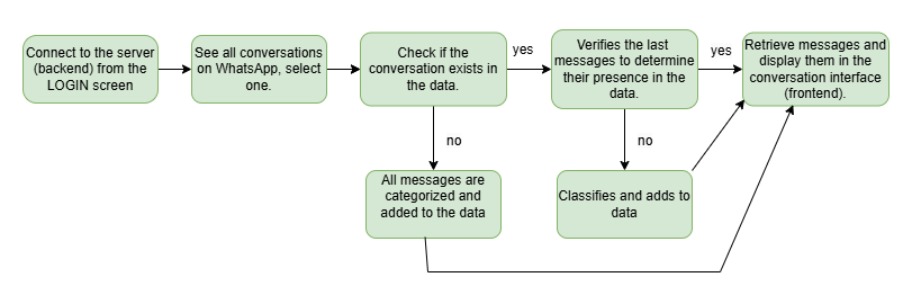
**9.1 DIAGRAMS**

**9.1.1 Package diagram**

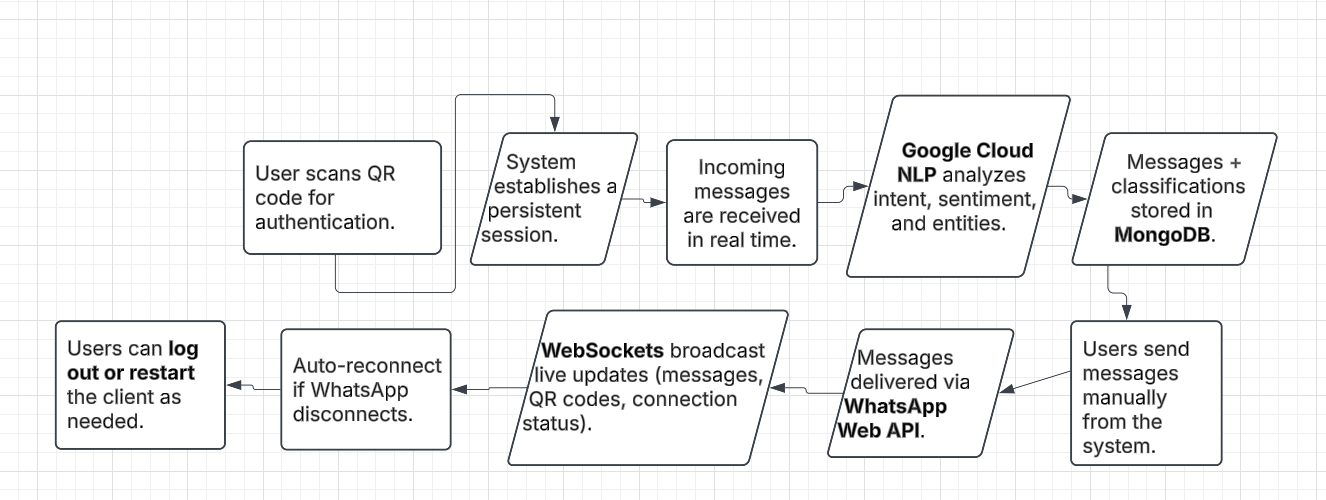
**תמונה שמכילה תרשים, תוכנית, שרטוט טכני, טקסט

התיאור נוצר באופן אוטומטי**

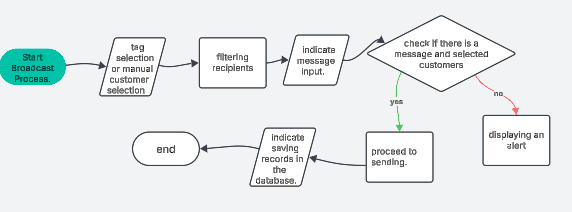
**9.1.2 FlowChart diagram**



WhatsApp Conversation Retrieval Process – This flowchart illustrates the process of connecting to the backend, selecting a WhatsApp conversation, verifying its presence in the database, classifying and storing messages, and retrieving them for display in the frontend conversation interface.



WhatsApp Integration System Workflow – This flowchart illustrates the end-to-end process of handling WhatsApp communication, including QR code authentication, real-time message reception, NLP-based classification, message storage, WebSocket updates, and session management to ensure seamless connectivity and automated processing.

****

This flowchart follows the logical order of a broadcast system, ensuring that only valid messages are sent to selected recipients.

**10. TEST PLANS**

**10.1. LOGIN TESTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Name | Description | Expected Results | Results | Comments |
| Test valid login credentials | Enter a valid username and password to test the login feature | The user is logged in successfully, and their role (admin/user) is identified correctly. | ✅ Successful login with correct role assignment. | Ensures correct authentication. |
| Test invalid login credentials | Enter an invalid username and password | The login feature rejects the credentials and displays an error message | ✅ Incorrect credentials were rejected with an error message. | Validates authentication security. |
| Test valid username, no password | Enter a valid username without entering a password. | The login feature rejects the credentials and displays an error message | ✅ Login attempt was blocked, error displayed. | Prevents blank password submissions. |
| Test no username, valid password | Enter a valid password but leave the username empty. | The login feature rejects the credentials and displays an error message | ✅ Login attempt was blocked, error displayed. | Ensures both fields are required. |
| Test session persistence | Log in and refresh the page. | The user remains logged in if a session/token is active. | ✅ Session persisted correctly after refresh. | Checks if authentication persists properly. |
| Test logout function | Click on the logout button after logging in. | The user is logged out and redirected to the login page. | ✅ Successfully logged out and redirected. | Ensures session is properly cleared. |

**10.2. MANAGER TESTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Name | Description | Expected Results | Results | Comments |
| Search customer by name | Use the search bar to find a specific customer. | The relevant customer appears in the results. | ✅ Search returned correct customer. | Ensures search functionality is accurate. |
| Add non-existing tag | Apply a new tag to a customer message. | The tag is successfully added. | ✅ Tag added without issues. | Tags should be unique and correctly applied. |
| Add existing tag | Attempt to add an already existing tag to a message. | The system prevents duplicate tags from being added. | ✅ Duplicate tag was blocked. | Avoids redundancy. |
| Add a tag with empty name | Try to create a tag without specifying a name. | The system does not allow the tag to be added. | ✅ Empty tag was not allowed. | Prevents invalid tag entries. |
| Filter messages by tag | Use a tag filter to display only messages associated with it. | Messages are correctly filtered based on the selected tag. | ✅ Messages were filtered properly. | Ensures messages are categorized correctly. |
| View customer chat history | Open a conversation with a customer. | Chat history loads correctly, displaying past messages. | ✅ Chat history displayed as expected. | Ensures message retrieval is functioning. |

**10.3. BROADCAST MESSAGE TESTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Name | Description | Expected Results | Results | Comments |
| Send a broadcast message | Select multiple customers and send a message. | The message is delivered to all selected recipients. | ✅ Messages sent successfully. | Ensures bulk messaging works. |
| Broadcast with empty message | Attempt to send a message without content. | The system prevents sending an empty message. | ✅ Empty message was blocked. | Avoids unnecessary API calls. |
| Broadcast with no recipients | Attempt to send a broadcast without selecting customers. | The system prevents sending and shows an error. | ✅ Sending was blocked with an error. | Ensures a valid audience selection. |
| Filter broadcast recipients by tag | Select a tag and check which customers appear in the list. | Only customers with the selected tag should appear. | ✅ Tag-based filtering worked correctly. | Ensures correct segmentation. |

**10.4. SALES & ANALYTICS TESTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Name | Description | Expected Results | Results | Comments |
| View sales insights | Sort and filter sales insights by category. | Sales data is correctly sorted and filtered. | ✅ Sorting and filtering worked properly. | Ensures accurate analytics processing. |
| View analytics dashboard | Open the dashboard to view customer engagement metrics. | The dashboard loads correctly with updated data. | ✅ Dashboard displayed up-to-date information. | Ensures real-time data analysis. |
| Group customers by interests | Use customer conversations to group them based on common topics. | Customers are grouped correctly according to chat content. | ✅ Customers were categorized correctly. | Ensures AI-driven customer segmentation. |

**10.5. WHATSAPP WEB INTEGRATION TESTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Name | Description | Expected Results | Results | Comments |
| Scan QR Code for login | Generate and scan the QR code to log in. | WhatsApp Web connects successfully. | ✅ QR code scanned, login successful. | Ensures seamless authentication. |
| Check WhatsApp connection status | Verify the client state after login. | The system displays "connected" or "disconnected" appropriately. | ✅ Connection status displayed correctly. | Prevents session-related issues. |
| Send a WhatsApp message | Manually send a message to a contact. | The message is delivered successfully. | ✅ Message was sent and received. | Ensures WhatsApp API functionality. |
| Receive WhatsApp messages | Send a test message to the connected WhatsApp. | The system receives and processes the message. | ✅ Incoming message was processed correctly. | Confirms real-time message reception. |
| WhatsApp Web session persistence | Stay logged in after a page refresh. | The session remains active if valid. | ✅ Session persisted without re-authentication. | Prevents frequent reauthentication. |
| WhatsApp logout test | Log out from WhatsApp Web via the system. | The system disconnects WhatsApp Web and requires a new QR code to log in again. | ✅ Logout worked as expected, QR reset. | Ensures secure session management. |

**11. Conclusion**

The ShopLINK project was developed to address the growing need for efficient customer interaction management via WhatsApp. As businesses increasingly rely on messaging platforms for support and marketing, traditional methods often become overwhelmed by the sheer volume of customer inquiries and promotional communications. To solve this, ShopLINK integrates WhatsApp Web, AI-powered Natural Language Processing (NLP), and a structured customer management system to automate and streamline communication.

Throughout the development of this project, we successfully implemented key features, including:

* WhatsApp Web integration with real-time message processing
* Google Cloud NLP-based message classification for identifying intent and sentiment
* Customer segmentation and management to optimize targeted communication
* Broadcast messaging system for sending promotional messages efficiently
* Scalable MongoDB storage for tracking customer interactions
* Socket.io-based real-time updates to ensure seamless communication between the backend and frontend

Developing ShopLINK came with several challenges, particularly in managing session persistence for WhatsApp Web, implementing accurate NLP classifications, and ensuring real-time message synchronization. Through rigorous testing and optimization, we refined our system to maintain high reliability and performance.

One of the most significant achievements of this project is the automated customer segmentation feature, which enables businesses to group customers based on chat content and engagement metrics. This ensures that marketing efforts are personalized and relevant, increasing customer engagement and satisfaction.

Despite our accomplishments, there are still areas for future improvement. Additional security enhancements, integration with payment systems, and multi-language support can further expand ShopLINK’s capabilities. Moreover, refining the AI-based clustering algorithm will allow for even more precise customer categorization.

In conclusion, ShopLINK successfully bridges the gap between businesses and their customers by leveraging AI-driven automation, real-time communication, and structured data management. It provides businesses with a scalable, efficient, and intelligent solution for managing their WhatsApp interactions, ultimately improving customer experience, engagement, and operational efficiency.

**12. User Documentation**

**12.1. User’s guide - Operating instruction**

**12.1.1 General description**

ShopLINK is an advanced communication system designed to integrate WhatsApp Web with customer management and AI-powered message classification. The system allows users to authenticate using a QR code, establishing a persistent session with WhatsApp Web to facilitate real-time messaging. Incoming messages are automatically processed using Google Cloud NLP, which analyzes their intent, sentiment, and key entities. These messages, along with their classifications, are then stored in MongoDB, enabling structured data retrieval and organization.

The platform offers an intuitive interface for managing customer conversations. Users can view all WhatsApp conversations, select specific chats, and check whether they exist in the system’s database. If a conversation is not yet recorded, the system automatically categorizes and stores the messages for future reference. For existing conversations, it verifies the latest messages to ensure all interactions are up to date before displaying them in the conversation interface.

Beyond message classification, ShopLINK provides a comprehensive customer management module. Users can add new customers, edit existing details, and delete outdated records while preserving conversation history. Additionally, the system includes a broadcast messaging feature that enables users to send bulk messages to selected recipients based on predefined tags or categories. This functionality ensures targeted and efficient communication with customers, enhancing engagement and marketing efforts.

Real-time updates are a critical component of the system. WebSockets broadcast live notifications for new messages, connection status, and QR code updates, ensuring a seamless user experience. The system is also designed to handle WhatsApp session disconnections by attempting automatic reconnection. If needed, users can manually log out or restart the client to restore connectivity.

The analytics dashboard provides insights into customer engagement, allowing businesses to track message trends, response rates, and customer interests. By grouping customers based on conversation topics, the system helps businesses tailor their outreach efforts more effectively.

ShopLINK is built to streamline WhatsApp-based interactions, improving response efficiency, enhancing data organization, and enabling AI-driven communication insights. Through its automated workflows, structured messaging, and real-time updates, it serves as a powerful tool for businesses aiming to manage customer conversations at scale.

**12.1.2 MANAGER SCREENS**

**General pages for all users:**

**תמונה שמכילה טקסט, צילום מסך, גופן, לוגו

התיאור נוצר באופן אוטומטי**

This is the login screen for ShopLINK. Users need to enter their username and password to access the system. We gave you the Username and the password.

תמונה שמכילה טקסט, צילום מסך, ירוק, לוגו

התיאור נוצר באופן אוטומטי

The welcome screen introduces users to ShopLINK, highlighting its main functionalities. It serves as an entry point before accessing the dashboard, analytics, product management, and broadcast messaging features.

**תמונה שמכילה צילום מסך, טקסט, תוכנה, דף אינטרנט

התיאור נוצר באופן אוטומטי**

This screen displays all WhatsApp conversations linked to ShopLINK. Users can search chats, filter messages by classification, and select a specific conversation to view its history and incoming messages in real-time.

תמונה שמכילה טקסט, צילום מסך, תוכנה, גופן

התיאור נוצר באופן אוטומטי The broadcast message interface allows users to send bulk messages to multiple customers. Users can filter recipients by tags, manually select customers, and type a message before sending. After clicking "Send Broadcast", a confirmation appears, ensuring the message has been delivered to the intended audience.

תמונה שמכילה טקסט, גופן, תרשים, תוכנה

התיאור נוצר באופן אוטומטי

This interface provides real-time analytics on customer engagement. Users can view messages sent over time, response rates, average response times, and popular engagement times. The analytics dashboard helps businesses track performance and optimize their communication strategy.

תמונה שמכילה טקסט, צילום מסך, תוכנה, עיצוב

התיאור נוצר באופן אוטומטיThe product catalog allows businesses to add, edit, or delete products. Users can enter details such as product name, price, category, and stock availability. This feature enables better product organization and helps businesses manage inventory more efficiently. **13.2 Maintenance**

## 13.2 Maintenance guide

### 13.2.1 App maintenance guide

In order to maintain the code, you first need to install a few tools and working environments on your computer.

Starting with:

1. VSCode - [Download Visual Studio Code - Mac, Linux, Windows](https://code.visualstudio.com/download) (Optional, can be any other environment that runs python files)
2. Google Cloud CLI - [install](https://cloud.google.com/natural-language/docs/sentiment-analysis-client-libraries)
3. After installing all these directories you have to clone the project from this repository in Github- <https://github.com/danvar249/CapstoneProject> , and then you need
4. Copy to frontend/src:

SERVER\_URL=http://localhost:5000

CLIENT\_URL=http://localhost:3000

TIMEOUT = 10000

1. Copy to backend:

DB\_USERNAME = danvar249

DB\_PASSWORD = 3foMj5eWTdThDwSi

PORT = 5000

1. Open a terminal and copy:

cd backend

npm i

1. Open a new terminal and copy:

cd frontend

npm i

Then you can run the code on your computer.

**14.REFERENCES**

1. Advantages & Disadvantages of Node.js : Why to Use Node.js?, <https://www.simform.com/blog/nodejs-advantages-disadvantages/>
2. AI <https://chatgpt.com/>
3. <https://cloud.google.com/natural-language/docs>
4. Express <https://expressjs.com/>
5. Interakt application <https://www.interakt.shop/>
6. Introduction to NodeJS <https://nodejs.dev/en/learn/>
7. React <https://react.dev/>
8. What is MERN Stack?  [“https://kenzie.snhu.edu/blog/what-is-mern-stack/”](file:///C:\Users\danva\AppData\Local\Microsoft\Windows\INetCache\IE\XRZFXBX2\)
9. What Is MongoDB? <https://www.mongodb.com/company/what-is-mongodb>
10. Why did we build Visual Studio Code? <https://code.visualstudio.com/docs/editor/whyvscode>
11. Zixflow application <https://zixflow.com/>
12. Zoko application <https://www.zoko.io/>