PROJECT REPORT

Submitted to

DEPARTMENT OF COMPUTER SCIENCE

VIVEKANAND EDUCATION SOCIETY'S

COLLEGE OF ARTS, SCIENCE AND COMMERCE,

SINDHI SOCIETY, CHEMBUR, MUMBAI 400071.

Alma Android Application

For Partial Fulfillment for Degree of

Bachelor of Science (Computer Science) Academic Year (2024-25)

COORDINATOR OF DEPARTMENT

Mr. Kamlakar Bhopatkar

COLLEGE GUIDE

Ms. Rajashree Date

SUBMITTED BY

Dhruv Pannalal Kushvaha

VIVEKANAND EDUCATION SOCIETY'S

COLLEGE OF ARTS, SCIENCE AND COMMERCE

Sindhi Society, Chembur, Mumbai 400071.

Phones: 25227514/25227470.

NAAC Re-Accredited 'A' Grade

CERTIFICATE

This is to certify that **Mr. Dhruv Pannalal Kushvaha** of T.Y.B.Sc (Computer Science) affiliated to University of Mumbai has successfully completed a project work entitled.

Alma Android Application

As partial fulfillment of the requirement for the degree of B.Sc. (Computer Science) for the academic year 2024-2025.

COORDINATOR OF DEPARTMENT	COLLEGE GUIDE
Mr. Kamlakar Bhopatkar	**Ms. Rajashree Date**

Date:

Examiner: College Seal

Acknowledgement:

It is with immense pride and gratitude that I present this project titled “**Alma Android Application.**” The journey of bringing this project to life has been one of profound learning, collaboration, and growth, and I am truly thankful to all those who have been a part of it.

First and foremost, I extend my heartfelt thanks to **V.E.S. College** and our esteemed Principal, **Dr. Anita Kanwar**, for providing a stimulating environment that nurtures innovation and academic excellence. Their constant encouragement and support have been vital in turning this vision into a reality.

I am deeply indebted to my project guide, **Mr. Kamlakar Bhopatkar**, whose invaluable mentorship and expertise have been the cornerstone of this project's success. His insightful guidance, encouragement, and unwavering support inspired me to explore new ideas and overcome challenges throughout the development process.

I would also like to express my sincere appreciation to my friends and peers for their unwavering support and encouragement. Their constructive feedback, shared enthusiasm, and willingness to assist at every stage were instrumental in shaping the **Alma Android Application** into what it is today.

This project is a testament to the power of perseverance, collaboration, and the pursuit of innovation. I feel truly privileged to have the opportunity to work with such an incredible mentor and support system.

INDEX

PRELIMINARY INVESTIGATION

Introduction

Children increasingly rely on AI chatbots for learning and guidance, yet most lack emotional intelligence, ethical safeguards, and age-awareness, making them unsuitable. Research by the World Economic Forum highlights risks like misinformation and inappropriate exposure, while the American Psychological Association (APA) stresses that children process information differently across ages, requiring context-aware AI responses. Our project, Alma: The Guardian Chatbot, ensures emotionally intelligent, safe, and age-appropriate conversations, acting as a companion and mentor rather than just a tool. By fostering trust, learning, and digital well-being, Alma provides personalized, ethical, and developmentally suitable AI interactions.

Description of Existing System

Existing AI chatbots such as ChatGPT, Google Assistant, and Alexa are widely used for general conversations, learning support, and entertainment. However, these chatbots are not designed specifically for children and lack features to ensure age-appropriate, emotionally intelligent, and ethically guided interactions. Most chatbots provide generic responses, failing to adapt to a child's cognitive development, emotional needs, or ethical considerations. While some educational chatbots exist, they primarily focus on academic learning rather than holistic child guidance.

Limitations of Existing System

- **Lack of Age Awareness**: Current AI chatbots do not modify responses based on a child’s age, leading to misinformation or inappropriate advice.
- **Absence of Emotional Intelligence**: These bots fail to engage with children in an empathetic and supportive manner, often giving detached or overly robotic responses.
- **No Ethical Safeguards**: AI chatbots lack protective measures to ensure children receive safe and developmentally suitable interactions, potentially exposing them to harmful or misleading content.

Description of Proposed System

Overview:

Alma is an AI-powered conversational assistant designed to provide age-appropriate, emotionally intelligent, and ethically guided interactions for children. Unlike generic AI chatbots, Alma adapts its responses based on the child's age group, ensuring safe, meaningful, and developmentally appropriate conversations. By acting as a responsible digital companion, Alma promotes learning, emotional well-being, and ethical decision-making while maintaining an engaging and friendly interaction experience.

Type of Application:

This project involves developing a mobile application for Android devices.

How the Application Works:

- **User Registration & Profile Setup:**
  - During signup, the **child enters their name and date of birth (DOB)** to determine their age group.
  - Once registered, they can start chatting with Alma immediately.
  - The profile stores the child's **name, age group, and chat history**.

- **Conversational Interaction:**
  - The chatbot customizes its **language, tone, and responses** based on the child's age.
  - If the child asks about **sensitive topics**, Alma provides **guidance in a child-friendly manner** rather than direct or inappropriate answers.
  - **Text-to-speech (TTS)** support allows the chatbot's responses to be read aloud.

- **Chat History:**
    - Every conversation is **automatically saved with a timestamp** in Firebase.
    - When the child opens their chat history, they can view **past conversations grouped by date**.

- **Module Descriptions:**
    A. **Login Module:**
        a. Secure login and authentication using Firebase Authentication.
        b. Options for email/password and Google sign-in.
    B. **Chat Module:**
        a. AI-driven chatbot with age-appropriate responses.
        b. TTS feature for voice-based interactions.
    C. **Chat History Module:**
        a. Displays previous conversations sorted by date.
        b. The child can scroll through past chats.

Advantages of Proposed System

Alma: The Guardian Chatbot offers a safe and personalized conversational experience for children, ensuring they receive age-appropriate, meaningful, and ethical responses. Unlike generic chatbots, Alma focuses on guiding children through various topics in a responsible manner while promoting digital well-being.

Advantages:

- **Age-Appropriate Conversations:**
  Alma tailors its responses based on the child's age group, ensuring safe and relevant interactions.
- **Privacy-Focused:**
  Chat history is stored securely and is only accessible to the child, maintaining their privacy.
- **User-Friendly Interface:**
  Simple and intuitive design, allowing children to engage easily without complex navigation.
- **Guidance Over Direct Answers:**
  Instead of providing direct or misleading answers to sensitive topics, Alma offers guidance in a supportive and educational manner.
- **Chat History Organization:**
  Past conversations are stored with timestamps, allowing children to revisit their chats in an organized way.

By fostering responsible digital interactions, Alma serves as a trusted AI companion that supports children's curiosity while ensuring a safe and structured chat environment.

Technologies Used

Languages:

1. **Kotlin**

- **Description**: Kotlin is a modern, statically-typed programming language that runs on the Java Virtual Machine (JVM) and can be used to develop Android applications.
- **Advantages:**
  - Fully interoperable with Java, allowing seamless integration with existing Java codebases.
  - Reduces boilerplate code, making the codebase more readable and maintainable.
  - Features like null safety help prevent common programming errors.

Database:

1. **Firebase**

- **Description**: Firebase is a flexible, scalable database for mobile, web, and server development. It is a NoSQL cloud database that allows you to store and sync data in real-time.
- **Advantages**:
  - Ensures that data is updated in real-time across all clients.
  - Allows data to be read and written even when offline, syncing changes when the device reconnects.
  - Automatically scales to handle large amounts of data and high traffic.

IDE:

1. **Android Studio**

- **Description**: Android Studio is the official IDE for Android development, based on IntelliJ IDEA.
- **Advantages**:
  - Includes everything needed for Android development, from code editing and debugging to performance profiling.
  - Seamlessly integrates with other Google tools and services, such as Firebase.

    - Regular updates ensure support for the latest Android features and technologies.

2. **Google Cloud Console**

- **Description**: Google Cloud Console is a web-based tool for managing Google Cloud resources and services.
- **Advantages**:
    - Provides a single interface for managing all Google Cloud services.
    - Offers real-time monitoring and analytics for your cloud resources.
    - Supports automation and scripting for streamlined management.

APIs:

1. **Google Gemini API**

- **Description**: Google Gemini API is a powerful AI model that enables applications to integrate conversational AI features.
- **Advantages**:
    - Provides intelligent responses based on user queries.
    - Capable of handling contextual understanding and personalization.
    - Scalable and easy to integrate within Android applications.

2. **Cloud Text to Speech API**

- **Description**: Google Cloud Text-to-Speech API converts text into natural-sounding speech using deep learning models.
- **Advantages**:
    - Supports multiple languages and voices.
    - Allows customization of pitch, speed, and volume.
    - Enhances accessibility by reading responses aloud.

SYSTEM ANALYSIS

Use Case Diagram

A use case diagram visually represents the various operations that a system performs, highlighting how different actors (users or other systems) interact with the system's functionality. It consists of use cases, actors, the system boundary, and the relationships between them. This diagram is an essential tool to understand the system's behavior from a user's perspective and to identify the interactions that take place within the system.

Elements of a Use Case Diagram

- **Actors:** An actor represents any entity (person, organization, or external system) that interacts with the system. The actor plays a specific role in the system and can initiate or participate in a sequence of events. For example, in the context of the **Alma** app, actors could include the **Child** (the user will interact with the chatbot). Actors are represented by stick figures and are positioned outside the system boundary.

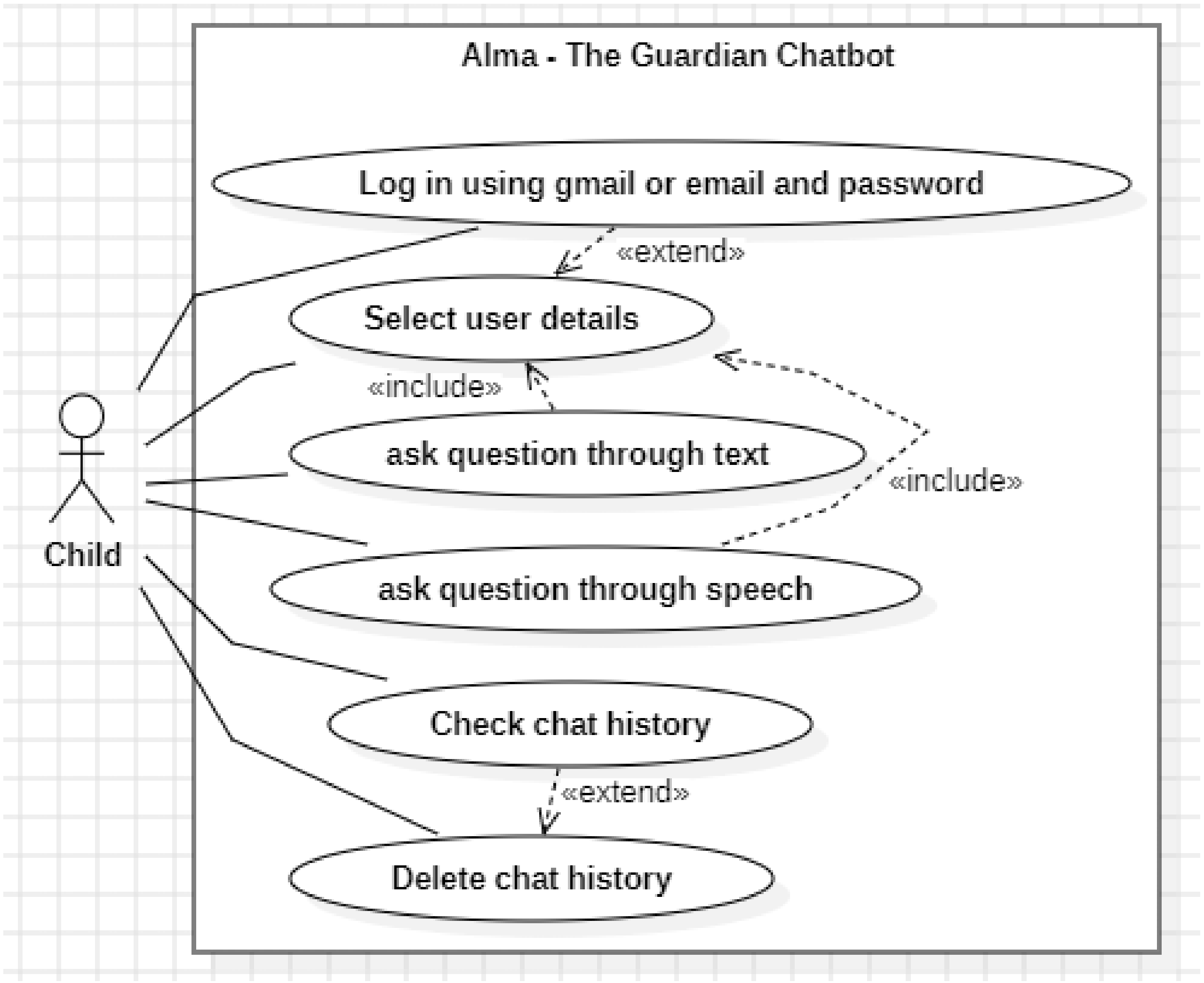
- **Use Cases:** A use case represents a distinct piece of functionality or behavior that the system performs in response to an actor's interaction. These use cases encapsulate the actions or services the system provides. For example, the "Select age group", "Clear old chats" and "Ask questions through text" of the Alma app are all individual use cases. Use cases are represented as ellipses within the system boundary.

- **System Boundary:** The system boundary defines the scope and limits of the system being modeled. It identifies what the system includes and excludes, helping to clearly demarcate the interactions between actors and the system. In the use case diagram, the system boundary is represented as a rectangle that encapsulates all the use cases.

- **Relationships:** Use cases can have different types of relationships, which help illustrate how they interact with each other:

1. **Extends:** This relationship indicates that a use case may have additional behavior under certain conditions. For instance, "Check chat history" might extend "Delete Chat History".
2. **Includes:** This relationship shows that one use case relies on another to complete its function. For example, "Get time limit warning" may include "Set app time limit" to function correctly, ensuring the alert window only appears when app limit is set.

Use case diagram


Alma - The Guardian Chatbot
Log in using gmail or email and password
«extend»
Select user details
«include»
ask question through text
«include»
ask question through speech
Check chat history
«extend»
Delete chat history
Child

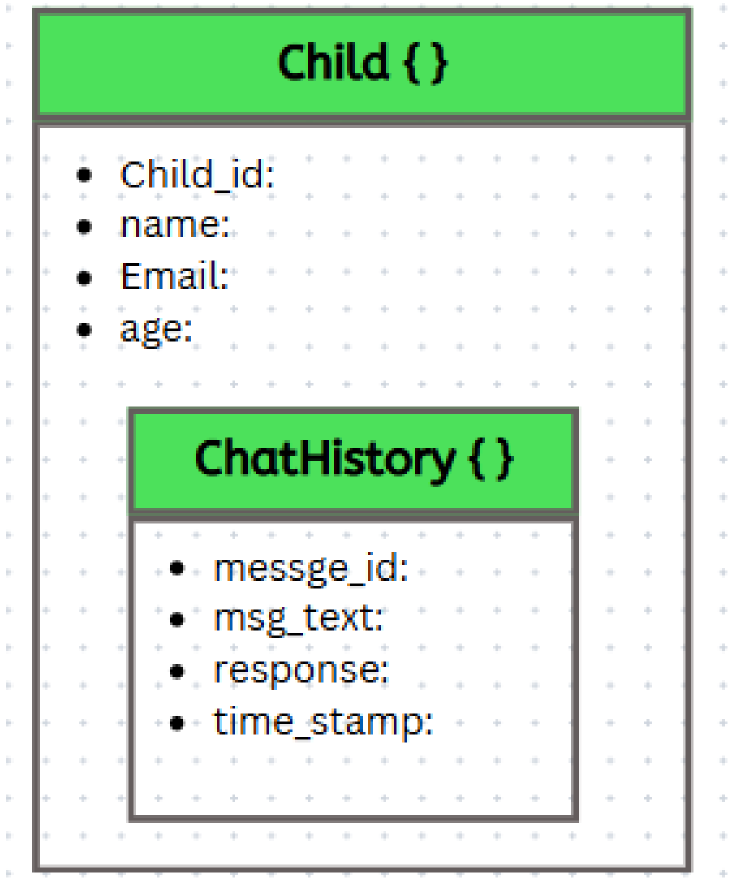
Database diagram

A **Database Diagram for NoSQL** that focuses solely on collections visually represents the structure of data within a NoSQL database. This type of diagram emphasizes how different collections are organized and their relationships without delving into the individual documents or fields. It provides a high-level view of the database schema, which is particularly useful in understanding how data is grouped and accessed.

Elements of a NoSQL Database Diagram:

- **Collections**: Collections are the primary building blocks of a NoSQL database. Each collection groups similar types of data and is schema-less, meaning it can contain documents with varying structures.

Database diagram



<u>Activity Diagram:</u>

An **Activity Diagram** visually represents the flow of activities in a system, illustrating the sequence of tasks, decisions, and interactions. This diagram is useful for understanding the dynamic aspects of a system by showing how processes are triggered, how they progress, and how they interconnect with other activities. It includes activities, decisions, transitions, swimlanes, and the start/end points, helping to map out the system's overall workflow and identify points of interaction or decision-making.

Elements of a Activity Diagram

- **Activities**: Activities represent the actions or tasks that need to be performed within the system. Each activity is depicted as a rounded rectangle. For example, in the context of the **Alma** app, activities could include **Login**, **Get Response** and **Send notification to Parent**. These activities form the core of the system's functionality, illustrating the steps users and the system take to complete tasks..

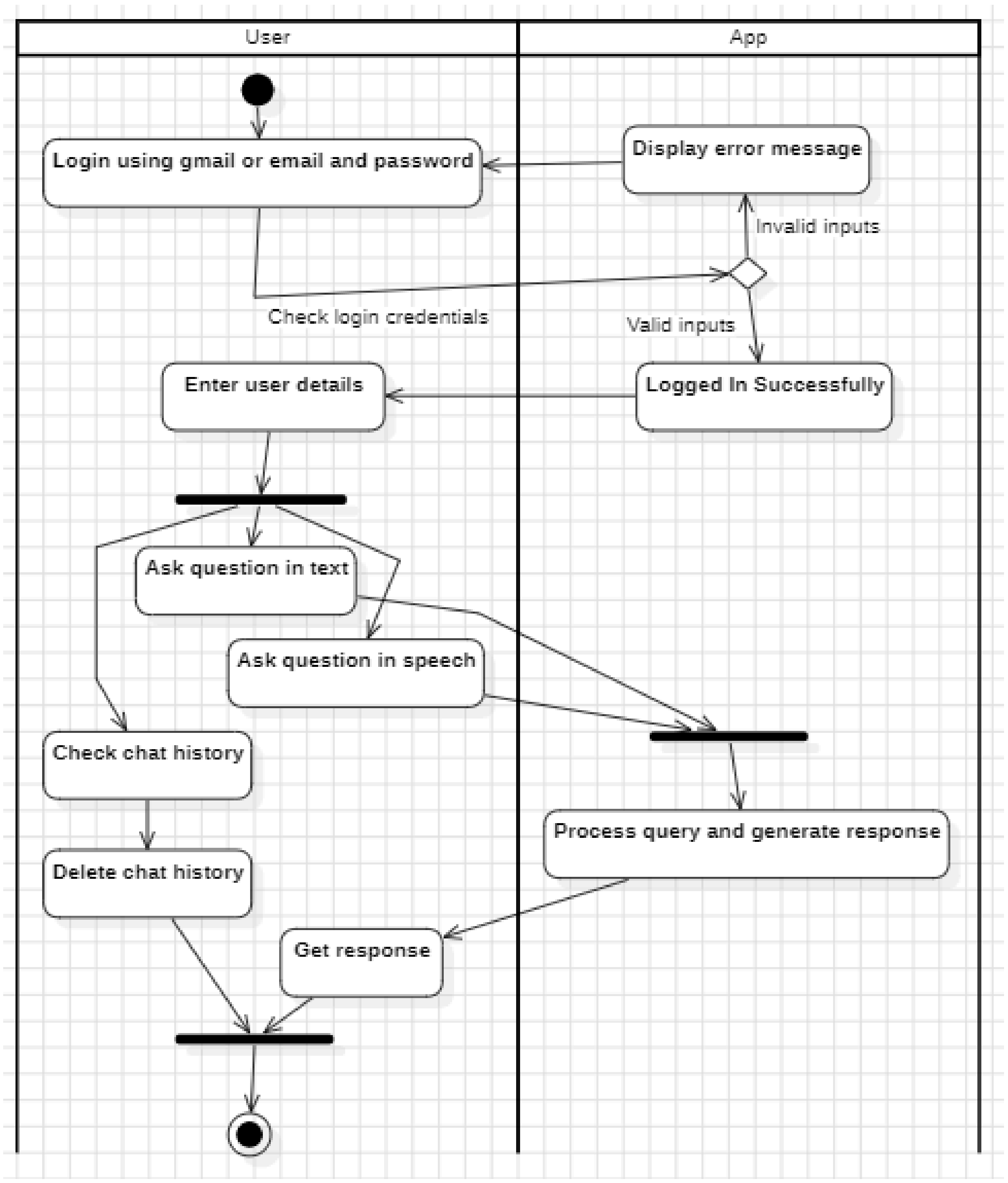
- **Initial Node**: The initial node represents the starting point of the workflow. It is shown as a solid black circle.

- **Final Node**: The final node represents the end of a workflow or process. It is depicted as a black circle with a surrounding outline.

- **Decision Node**: A decision node represents a branching point in the flow, where different paths can be taken based on a condition. It is depicted as a diamond shape. A decision node could be used when **Checking if Geofence is valid**, leading to different outcomes depending on whether the child has left the geofenced area.

- **Swimlanes**: Swimlanes are vertical or horizontal sections that divide the activities among different actors or system components. There could be swimlanes for the **Parent**, **Child**, and App.

Activity Diagram:



<u>Sequence Diagram</u>

A **Sequence diagram** models the interactions between objects in the system over time. It shows how objects (or components) communicate with each other, the sequence of events that occur, and the messages that are exchanged. This diagram helps in visualizing the sequence of interactions within a specific use case or feature, highlighting the flow of control between various system components and actors.

Elements of a Sequence Diagram

- **Actors**: Actors are entities (users or external systems) that interact with the system. In a sequence diagram, actors are represented by a rectangle with a label, placed at the top.

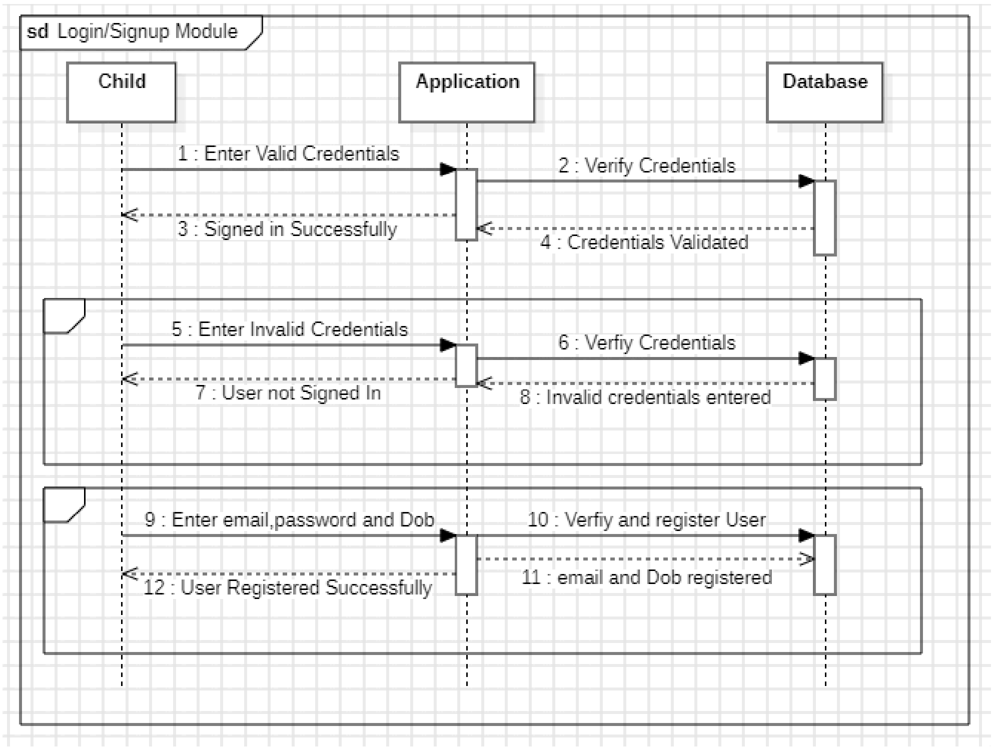
- **Lifelines**: A lifeline represents an individual participant in the interaction, which could be an actor or a system object. It is shown as a vertical dashed line below the actor or object. The length of the lifeline indicates the duration of the participant's involvement in the interaction. Lifelines could exist for **Child, Database** and **Application**.

- **Messages**: Messages are the arrows that show the interaction between lifelines. They represent the communication or method calls that occur between objects or actors. Messages can be synchronous (requiring a response) or asynchronous (no immediate response needed). A message could be sent from the **User** to **Application** to **Enter date of birth**.

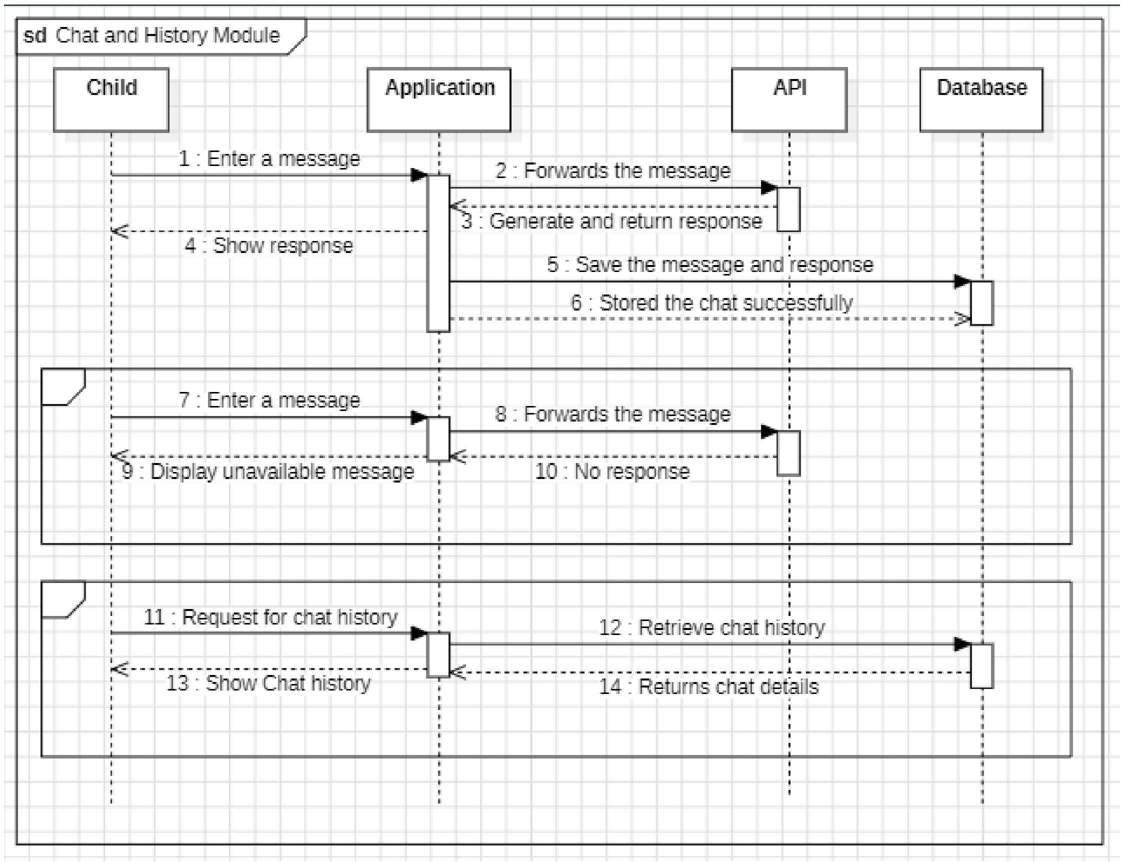
- **Return Messages:** Return messages show responses from the called object back to the sender. These are represented as dashed arrows.

Sequence Diagram

Login and Sign Up:


sd Login/Signup Module
Child
Application
Database
1 : Enter Valid Credentials
2 : Verify Credentials
3 : Signed in Successfully
4 : Credentials Validated
5 : Enter Invalid Credentials
6 : Verfiy Credentials
7 : User not Signed In
8 : Invalid credentials entered
9 : Enter email,password and Dob
10 : Verfiy and register User
11 : email and Dob registered
12 : User Registered Successfully

Chat and History Module:


sd Chat and History Module
Child
Application
API
Database
1 : Enter a message
2 : Forwards the message
3 : Generate and return response
4 : Show response
5 : Save the message and response
6 : Stored the chat successfully
7 : Enter a message
8 : Forwards the message
9 : Display unavailable message
10 : No response
11 : Request for chat history
12 : Retrieve chat history
13 : Show Chat history
14 : Returns chat details

Deployment Diagram

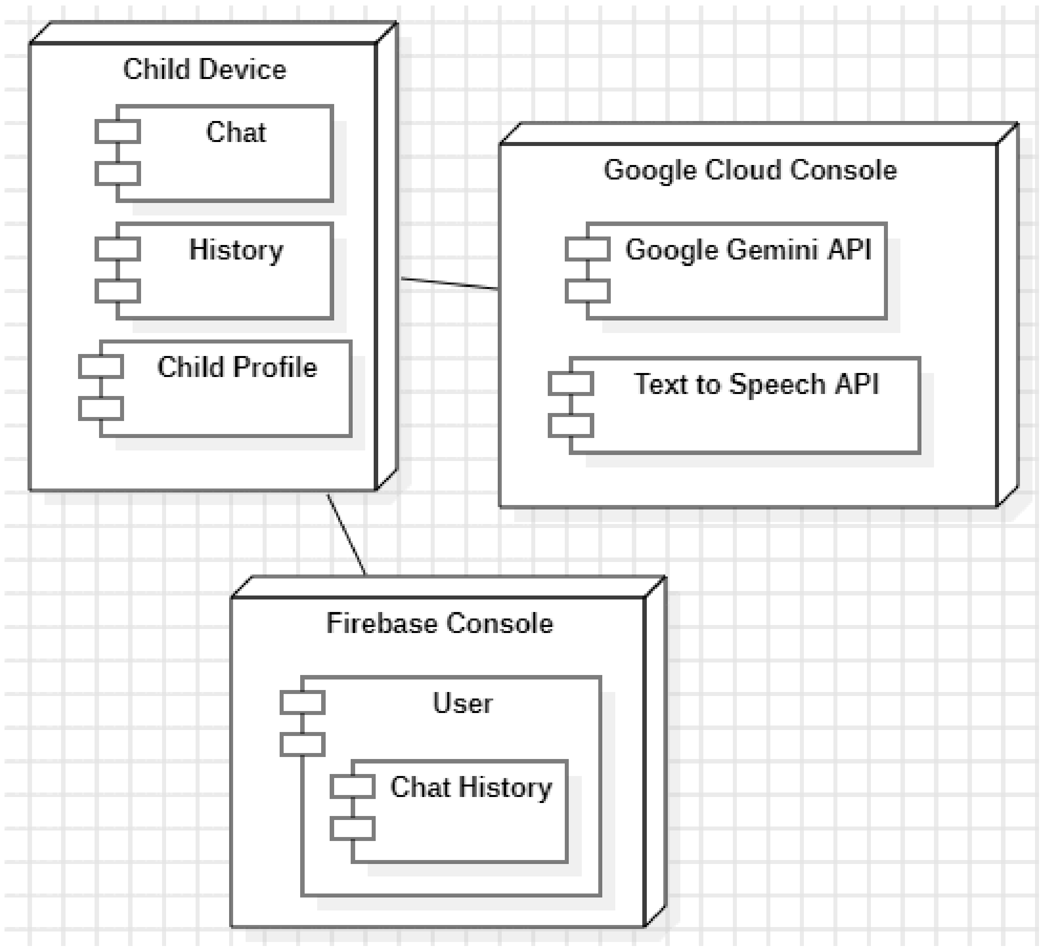
A **Deployment Diagram** visualizes the physical deployment of artifacts (software components) on hardware nodes in a system. It helps represent how software runs on hardware, showing the physical architecture of the system.

Elements of a Deployment Diagram

- **Nodes**: Nodes represent the physical or virtual hardware (like devices, servers, or environments) where software is deployed. In a deployment diagram, they are represented as 3D cubes. For example: Firebase Database and Mobile Device

- **Components:** Components are the software modules or parts that are deployed on nodes. For example: Chat and History.

Deployment Diagram



SYSTEM IMPLEMENTATION

Test Cases

Screenshots:

Bibliography & References