*A project report on*

**CHATBOT: The Virtual Agent**

*Submitted in partial fulfillment for the award of the degree of*

Bachelor of Technology in

Computer Science and Engineering

*by*

**Shivam Anand (18BCE0490)**



# SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

June 2022

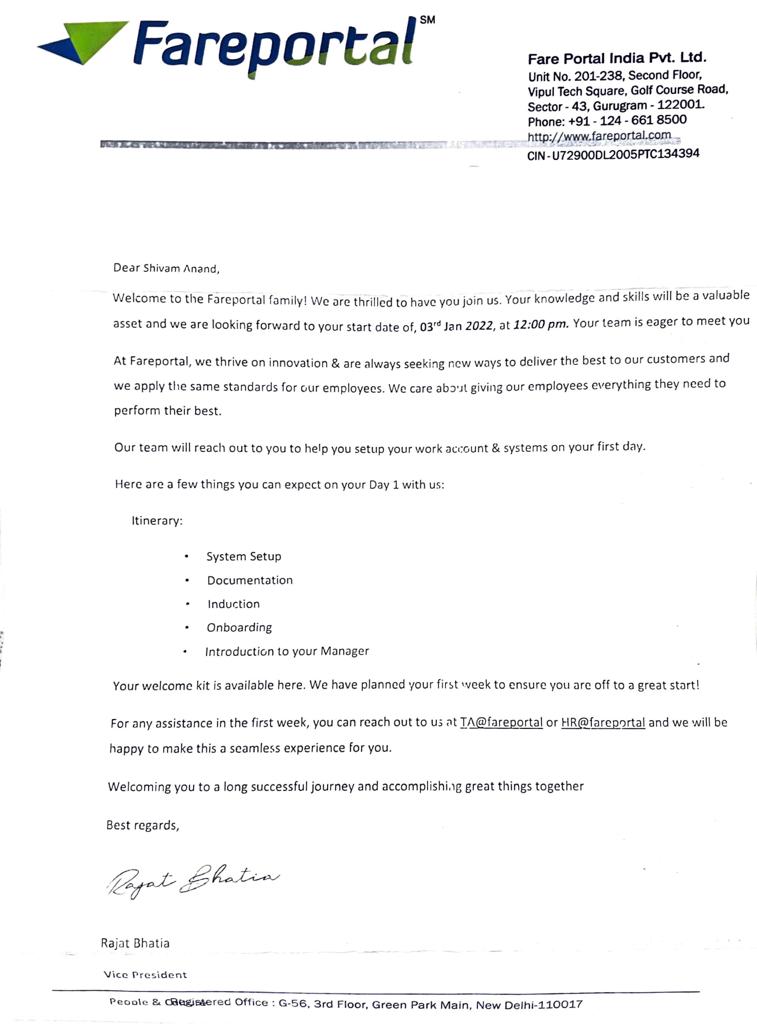
## DECLARATION

I hereby declare that the thesis entitled **“CHATBOT: The Virtual Agent”** submitted by me to Vellore Institute of Technology, Vellore, for the award of the degree of **Bachelors in Technology** in **Computer Science and Engineering** is a record of bonafide work carried out by me under the supervision of Mr. Digvijay Panwar.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Gurugram

Date:10/06/2022 Signature of the Candidate



## ABSTRACT

The “**Chatbot**” is a web-based project which is made customer support through chat over the Intermate. In this project I tried to show the working of Web- Based application and cover the basic functionality of a Automated “Chatbot”.

Chatbot can be as simple as rudimentary programs that answer a simple query with a single-line response, or as sophisticated as digital assistants that learn and evolve to deliver increasing levels of personalization as they gather and process information.

The way of conversation is now completely changed with the help of Chatbot without much use of customer care executives for selling and customer support through Internet.

Science has made great advancements to enhance the interaction between humans and machines by leaps and bounds. A Chatbot is artificial intelligence software that can simulate a conversation with a user in natural language. Artificial Intelligence gives a human touch to every conversation the chatbot strikes. The bot understands the user’s query and triggers an accurate response. Chatbot with AI powers makes your bot capable and intelligent to answer complex queries. Smart interactions save the user’s time by helping them to find the right information and address their queries. Especially in big data applications this reduces the computational burden, achieving faster iterations in trade for a slightly lower convergence rate. With the help of this algorithm, we will be contributing to the betterment of the society.

## ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to Dr G. Viswanathan, Chancellor, VIT University for creating an environment conducive for learning and growth academically as well as in extracurricular activities that enable every student to reach greater heights.

I would like to thank Dr. S Vairamuthu, HOD School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, for always encouragement and support, without which this project and the last four years would not have been possible.

I would also like to thank all the faculty and laboratory staff of the Department of Computer Science and Engineering.

I would like to express my gratitude to Rajat Bhatia SVP, People & Culture Fareportal India Pvt. Ltd for giving me an internship opportunity at Fareportal India Pvt. Ltd. Additionally, I would like to thank my managers Digvijay Panwar, Abhishek Yogi and team Ayush Singh & Himanshu for guiding me throughout the internship.

Lastly, I would like to thank the entire Operations team at Fareportal India Pvt. Ltd. for welcoming me into the organization and helping me tirelessly during every point in the internship.

It is indeed a pleasure to thank my family and friends who persuaded and encouraged me to take up and complete this task. At last but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place: Gurugram, India

Date:10/06/2022 **Shivam Anand**

# CONTENTS

[ABSTRACT iv](#_TOC_250013)

ACKNOWLEDGEMENTS v

[CONTENTS vi](#_TOC_250012)

[LIST OF FIGURES viii](#_TOC_250011)

[LIST OF ACRONYMS ix](#_TOC_250010)

CHAPTER 1 INTRODUCTION

* 1. [ABOUT THE COMPANY 1](#_TOC_250009)
  2. [PRODUCTS 2](#_TOC_250008)

CHAPTER 2 INTRODUCTION TO THE PROJECT

* 1. [Features of Chatbot 4](#_TOC_250007)
  2. [Brief Overview of Technology 6](#_TOC_250006)
  3. TOOLS USED

CHAPTER 3 OBJECTIVES OF THE PROJECT

* 1. [SYSTEM ANALYSIS 7](#_TOC_250005)
  2. [EXISTING SYSTEM 9](#_TOC_250004)
  3. [PROPOSED SYSTEM 18](#_TOC_250003)

[CHAPTER 4 FEASIBILITY](#_TOC_250002) STUDY

* 1. ECONOMIC FEASIBILITY 28

4.2 TECHNICAL FEASIBILITY 31

* 1. [OPERATIONAL FEASIBILITY 32](#_TOC_250001)

CHAPTER 5 Design Strategy

* 1. HARDWARE REQUIREMENT 35
  2. SOFTWARE REQUIREMENT 35

CHAPTER 6 DATA FLOW DIAGRAM

* 1. ELEMENTS OF DFD 35

6.2 PROCESS ENTITY 35

6.3 DATA FLOW ENTITY 35

6.4 DATA STORE ENTITY 35

6.5 EXTERNAL ENTITY 35

6.6SYMBOLS USED IN DFD 35

6.7 LEVEL 0 DFD 35

6.8 LEVEL 1 DFD 35

6.9 LEVEL 2 DFD 35

6.10 E-R DIAGRAM 35

CHAPTER 7 MODULE DESCRIPTION

7.1 MODULE DESCRIPTION 35

CHAPTER 8 CONCLUSIONS AND REFERENCES

8.1 CONCLUSION 35

8.2 REFERENCES 35

**List of Figures**

[Figure 1: Company Logo - 1 -](#_Toc73298246)

[Figure 2: Logo of HTML - 5 -](#_Toc73298247)

[Figure 3: Logo of CSS - 6 -](#_Toc73298248)

[Figure 4: Logo of JavaScript - 8 -](#_Toc73298249)

[Figure 5: Logo of React - 9 -](#_Toc73298250)

[Figure 6: Logo of Angular JS - 12 -](#_Toc73298251)

[Figure 7: Logo of Node.js - 18 -](#_Toc73298252)

[Figure 8: Logo of TypeScript - 19 -](#_Toc73298253)

[Figure 9: Logo of C# - 20 -](#_Toc73298254)

[Figure 10: Logo of . NET - 21 -](#_Toc73298255)

[Figure 11: Logo of MYQSL - 22 -](#_Toc73298256)

[Figure 12: Logo of Dialogflow - 22 -](#_Toc73298257)

[Figure 13: Basic Flow for Intent Matching and responding to the end-user - 23 -](#_Toc73298258)

[Figure 14: Example that uses Context - 23 -](#_Toc73298259)

[Figure 15: Logo of Visual Studio Code - 24 -](#_Toc73298260)

[Figure 16: Logo of Microsoft Azure - 24 -](#_Toc73298261)

[Figure 17: Logo of Postaman - 25 -](#_Toc73298262)

[Figure 18: Logo of Google Cloud Platform - 25 -](#_Toc73298263)

[Figure 19: Level 0 Dfd - 26 -](#_Toc73298264)

[Figure 20: Level 1 Dfd - 26 -](#_Toc73298265)

[Figure 21: Level 2 Dfd - 27 -](#_Toc73298266)

[Figure 22: E-R Diagram - 27 -](#_Toc73298267)

[Figure 23: Flow Diagram - 28 -](#_Toc73298268)

[Figure 24: code for majority element - 28 -](#_Toc73298269)

[Figure 25: Landing Page of Chatbot - 29 -](#_Toc73298270)

[Figure 26:Initial Message from Chatbot - 29 -](#_Toc73298271)

[Figure 27:Chatbot Interaction Page - 30 -](#_Toc73298272)

**LIST OF ACRONYMS**

1. PaaS: Platform as a Service

2. SQL: Structured Query Language

3. RDBMS: Relational Database Management System

4. GCP: Google Cloud Platform

5. CI/CD: Continuous Integration/ Continuous Deployment

6. API: Application Programming Interface

7. NPM: Node Packet Manager

8. VM: Virtual Machine

9. JS: Java Script

**Chapter 1**

**Introduction**

* 1. **About the Company**



Figure 1 Company Logo

Fareportal is a technology company that powers leading hybrid travel agencies like CheapOair, OneTravel, and Travelong, a veteran corporate travel agency founded in Morristown, New Jersey in 1976. our strong knowledge and experience in the air travel booking vertical, including corporate travel management and automation of wholesale distribution of complex international airfares, today we operate in North America, Europe, and Asia. We’re also proud to partner with over 500 airlines, over 1 million hotels, and hundreds of car agencies worldwide.

Fareportal’s unique hybrid business model bridges the gap between an online travel agency and a traditional travel agency by providing a convenient online booking capability as well as a 24/7 personalized trip booking experience arranged by hundreds of trained and certified travel agents in multiple countries and in multiple languages. By leveraging - 5 -company owned and operated contact centers located worldwide, we are able to deliver one of the highest levels of customer service and support in the air travel business.

Travelers around the world can find and book their perfect trip on our websites, mobile and tablet apps, and by calling one of our hundreds of trained and certified travel agents.

Fareportal’s partner airlines benefit from the broad customer reach and strong customer value proposition as compared to the pure OTA model. Focused more on selling international and higher yield complex airfares, partner airlines obtain higher revenue on a per seat basis.

Fareportal’s constant innovation allows partner airlines to benefit from selling ancillaries on the booking path thereby further optimizing their revenues and making them more predictable. This also helps the Airlines in streamlining airport operations, having less people wait in line to pay for preferred seats and bags. Read more about how Fareportal has been first to sell ancillaries for American Airlines, Air Canada and Spirit Airlines.

* 1. **Products: -**

**CheapOair**

CheapOair is the smartest way for savvy travelers to book their next trip. CheapOair allows customers to compare and book flights on over 450 airlines online, on award-winning mobile apps, or by phone with live travel agents.

**OneTravel**

OneTravel simplifies the travel shopping experience by combining value-driven deals on flights, hotels, cars and vacation packages with 24/7 customer care only one mouse click or phone call away.

**Fareportal Media Group**

Fareportal Media Group is the independent media division that manages the advertising operations, sales and business development for Fareportal’s travel websites. Through custom, integrated solutions, Fareportal ensures that its advertising partners get the best return on their investment.

**Travelong**

Travelong is a full-service travel agency serving more than 100 corporate clients and 30 home-based agents across the United States. Established in 1933, Travelong’s innovative and highly experienced travel consultants have provided client satisfaction for over eight decades.

**Royal Scenic**

Royal Scenic is a national wholesale supplier, air consolidator and preferred Fareportal partner with offices in Vancouver, Toronto and Montreal. Royal Scenic offers quality products and customer service exclusively to the retail travel community for both corporate and leisure markets.

**Dukes Court Travel Ltd.**

Duke’s Court Travel is based in London and offers a wide range of travel services. Established in 1989, Duke’s Court Travel is an appointed agency for over 50 major international airlines. Licensed by the UK Civil Aviation Authority under ATOL, it offers full financial protection to any who book directly through the agency

**Chapter-2**

**INTRODUCTION OF THE PROJECT**

**Chatbot**

**Use of Chatbots**

Chatbots are used in dialog systems for various purposes including customer service, request routing, or information gathering. While some chatbot applications use extensive word-classification processes, natural-language processors, and sophisticated AI others simply scan for general keywords and generate responses using common phrases obtained from an associated library or database.

**What is “Chatbot?**

A **chatbot** is a software application used to conduct an on-line chat conversation via text or text-to-speech, in lieu of providing direct contact with a live human agent. A chatBot is a type of software that can help customers by automating conversations and interact with them through messaging platforms.

**Administrator** of “Chatbot” has multiple features such as Add, Delete, Update the training data sets and user response.

By contrast, chatbots allow businesses to engage with an unlimited number of customers in a personal way and can be scaled up or down according to demand and business needs. By using chatbots, a business can provide humanlike, personalized, proactive service to millions of people at the same time

Chatbots can increase customer satisfaction through frictionless support. It can offer precise answers to questions and point customers in the right direction. Moreover, smart chatbots can offer contextual advice and escalate conversations to a live agent when necessary

**2.1 Features of “ChatBot”.**

1. Save responding time
2. Available anytime and anywhere
3. Improve customer engagement
4. Personalize conversations every time
5. Nudge towards actions
6. Quicker resolutions and comfortable communication
7. It is also affordable

**2.2 Brief overview of the technology:**  
Front end: HTML, CSS, JavaScript, React.

1. HTML: HTML is used to create and save web document. E.g. Notepad/Notepad++
2. CSS : (Cascading Style Sheets) Create attractive Layout
3. JavaScript: it is a programming language, commonly use with web browsers.
4. React is a declarative, efficient, and flexible javascript  library for building user interfaces. It lets you compose complex UIs from small and isolated pieces

**Back end: Node.js, TypeScript , MySQL**

1. 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
2. MySQL: MySql is a database, widely used for accessing querying, updating, and managing data in databases.
3. TypeScript: TypeScript is a programming language developed and maintained by Microsoft. It is a strict syntactical superset of JavaScript and adds optional static typing to the language. It is designed for the development of large applications and transpiles to JavaScript.
4. C#: **it** a simple, modern, general-purpose, object-oriented programming language developed by Microsoft within its .NET initiative led by Anders Hejlsberg.**C#** is one of the most popular programming languages and can be used for a variety of things
5. Dialogflow ES: Dialogflow is a natural language understanding platform that makes it easy to design and integrate a conversational user interface into your mobile app, web application, device, bot, interactive voice response system, and so on. Using Dialogflow you can provide new and engaging ways for users to interact with your product.

**2.3 Tools Used: -**

**2.3.1 HTML**



Figure 2: Logo of HTML

The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input /> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behaviour and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

**2.3.2 CSS**



Figure 3: Logo of CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTMLCSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.[3] This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate.css file, which reduces complexity and repetition in the structural content; and enable the.css file to be cached to improve the page load speed between the pages that share the file and its formatting.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.[4]

The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

**2.3.2 JavaScript**

****

Figure 4: Logo of JavaScript

JavaScript often abbreviated JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. Over 97% of websites use JavaScript on the client side for web page behaviour, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users' devices.

JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

The ECMAScript standard does not include any input/output (I/O), such as networking, storage, or graphics facilities. In practice, the web browser or other runtime system provides JavaScript APIs for I/O.

JavaScript engines were originally used only in web browsers, but are now core components of some servers and a variety of applications. The most popular runtime system for this usage is Node.js.

During the period of Internet Explorer dominance in the early 2000s, client-side scripting was stagnant. This started to change in 2004, when the successor of Netscape, Mozilla, released the Firefox browser. Firefox was well received by many, taking significant market share from Internet Explorer. Meanwhile, very important developments were occurring in open-source communities not affiliated with ECMA work. In 2005, Jesse James Garrett released a white paper in which he coined the term Ajax and described a set of technologies, of which JavaScript was the backbone, to create web applications where data can be loaded in the background, avoiding the need for full page reloads. This sparked a renaissance period of JavaScript, spearheaded by open-source libraries and the communities that formed around them. Many new libraries were created, including jQuery, Prototype, Dojo Toolkit, and MooTools.

**2.3.4 React**

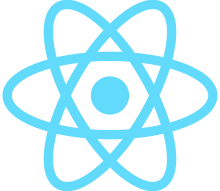


Figure 5: Logo of React

React (also known as React.js or ReactJS) is a free and open-source front-end JavaScript library for building user interfaces based on UI components. It is maintained by Meta (formerly Facebook) and a community of individual developers and companies. React can be used as a base in the development of single-page, mobile, or server-rendered applications with frameworks like Next.js. However, React is only concerned with state management and rendering that state to the DOM, so creating React applications usually requires the use of additional libraries for routing, as well as certain client-side functionality.

React code is made of entities called components. These components are reusable and must be formed in the SRC folder following the Pascal Case as its naming convention (capitalize camelCase). Components can be rendered to a particular element in the DOM using the React DOM library. When rendering a component, one can pass the values between components through "props":

Lifecycle methods for class-based components use a form of hooking that allows the execution of code at set points during a component's lifetime.

* shouldComponentUpdate allows the developer to prevent unnecessary re-rendering of a component by returning false if a render is not required.
* componentDidMount is called once the component has "mounted" (the component has been created in the user interface, often by associating it with a DOM node). This is commonly used to trigger data loading from a remote source via an API.
* componentWillUnmount is called immediately before the component is torn down or "unmounted". This is commonly used to clear resource-demanding dependencies to the component that will not simply be removed with the unmounting of the component (e.g., removing any setInterval () instances that are related to the component, or an "eventListener" set on the "document" because of the presence of the component)
* Render is the most important lifecycle method and the only required one in any component. It is usually called every time the component's state is updated, which should be reflected in the user interface.

JSX, or JavaScript Syntax Extension, is an extension to the JavaScript language syntax. Similar in appearance to HTML, JSX provides a way to structure component rendering using syntax familiar to many developers. React components are typically written using JSX, although they do not have to be (components may also be written in pure JavaScript). JSX is similar to another extension syntax created by Facebook for PHP called XHP.

Hooks are functions that let developers "hook into" React state and lifecycle features from function components. Hooks do not work inside classes — they let you use React without classes.

React provides a few built-in hooks like useState, useContext, useReducer , useMemo and useEffect. Others are documented in the Hooks API Reference. useState and useEffect, which are the most commonly used, are for controlling state and side effects respectively.

**2.3.5 Angular JS**



Figure 6: Logo of Angular JS

AngularJS was a JavaScript-based open-source front-end web framework for developing single-page applications. It was maintained mainly by Google and a community of individuals and corporations. It aimed to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–view model (MVVM) architectures, along with components commonly used in web applications and progressive web applications.

The AngularJS framework worked by first reading the Hypertext Markup Language (HTML) page, which had additional custom HTML attributes embedded into it. Angular interpreted those attributes as directives to bind input or output parts of the page to a model that is represented by standard JavaScript variables. The values of those JavaScript variables could be manually set within the code or retrieved from static or dynamic JSON resources.

AngularJS was built on the belief that declarative programming should be used to create user interfaces and connect software components, while imperative programming was better suited to defining an application's business logic. The framework adapted and extended traditional HTML to present dynamic content through two-way data-binding that allowed for the automatic synchronization of models and views. As a result, AngularJS de-emphasized explicit Document Object Model (DOM) manipulation with the goal of improving testability and performance.

AngularJS's design goals included:

* to decouple DOM manipulation from application logic. The difficulty of this is dramatically affected by the way the code is structured.
* to decouple the client side of an application from the server-side. This allows development work to progress in parallel and allows for reuse of both sides.
* to provide structure for the journey of building an application: from designing the UI, through writing the business logic, to testing.

AngularJS implemented the MVC pattern to separate presentation, data, and logic components. Using dependency injection, Angular brought traditionally server-side services, such as view-dependent controllers, to client-side web applications. Consequently, much of the burden on the server could be reduced.

**2.3.6 Node.JS**



Figure 7: Logo of 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.

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

Node.js allows the creation of Web servers and networking tools using JavaScript and a collection of "modules" that handle various core functionalities. Modules are provided for file system I/O, networking (DNS, HTTP, TCP, TLS/SSL, or UDP), binary data (buffers), cryptography functions, data streams, and other core functions. Node.js's modules use an API designed to reduce the complexity of writing server applications.

JavaScript is the only language that Node.js supports natively, but many compile-to-JS languages are available. As a result, Node.js applications can be written in Coffee Script, Dart, TypeScript, ClojureScript and others.

Node.js is primarily used to build network programs such as Web servers The most significant difference between Node.js and PHP is that most functions in PHP block until completion (commands execute only after previous commands finish), while Node.js functions are non-blocking (commands execute concurrently or even in parallel, and use call-backs to signal completion or failure).

Node.js is officially supported on Linux, macOS and Microsoft Windows 8.1 and Server 2012 (and later), with tier 2 support for SmartOS and IBM AIX and experimental support for FreeBSD. OpenBSD also works, and LTS versions available for IBM i (AS/400).[53] The provided source code may also be built on similar operating systems to those officially supported or be modified by third parties to support others such as NonStop OS and Unix servers.

**2.3.7 TypeScript**



Figure 8: Logo of TypeScript

TypeScript is a programming language developed and maintained by Microsoft. It is a strict syntactical superset of JavaScript and adds optional static typing to the language. It is designed for the development of large applications and trans piles to JavaScript. As it is a superset of JavaScript, existing JavaScript programs are also valid TypeScript programs.

TypeScript may be used to develop JavaScript applications for both client-side and server-side execution (as with Node.js or Deno). Multiple options are available for transpiration. The default TypeScript Compiler can be used, or the Babel compiler can be invoked to convert TypeScript to JavaScript.

TypeScript supports definition files that can contain type information of existing JavaScript libraries, much like C++ header files can describe the structure of existing object files. This enables other programs to use the values defined in the files as if they were statically typed TypeScript entities. There are third-party header files for popular libraries such as jQuery, MongoDB, and D3.js. TypeScript headers for the Node.js basic modules are also available, allowing development of Node.js programs within TypeScript.

TypeScript originated from the shortcomings of JavaScript for the development of large-scale applications both at Microsoft and among their external customers. Challenges with dealing with complex JavaScript code led to demand for custom tooling to ease developing of components in the language.

TypeScript is a strict superset of ECMAScript 2015, which is itself a superset of ECMAScript 5, commonly referred to as JavaScript. As such, a JavaScript program is also a valid TypeScript program, and a TypeScript program can seamlessly consume JavaScript. By default the compiler targets ECMAScript 5, the current prevailing standard, but is also able to generate constructs used in ECMAScript

**2.3.8 C#**

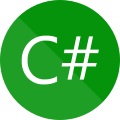


Figure 9: Logo of C#

C# is a general-purpose, multi-paradigm programming language. C# encompasses static typing, strong typing, lexically scoped, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines. C# is a simple, modern, general-purpose, object-oriented programming language developed by Microsoft within its .NET initiative led by Anders Hejlsberg. C# is one of the most popular programming languages and can be used for a variety of things.it is a general-purpose, multi-paradigm programming language. C# encompasses static typing, strong typing, lexically scoped, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines.

By design, C# is the programming language that most directly reflects the underlying Common Language Infrastructure (CLI). Most of its intrinsic types correspond to value-types implemented by the CLI framework. However, the language specification does not state the code generation requirements of the compiler: that is, it does not state that a C# compiler must target a Common Language Runtime, or generate Common Intermediate Language (CIL), or generate any other specific format

**2.3.9 .NET**



Figure 10: Logo of .NET

The .NET Framework (pronounced as "dot net") is a proprietary software framework developed by Microsoft that runs primarily on Microsoft Windows. It was the predominant implementation of the Common Language Infrastructure (CLI) until being superseded by the cross-platform .NET project. It includes a large class library called Framework Class Library (FCL) and provides language interoperability (each language can use code written in other languages) across several programming languages. Programs written for .NET Framework execute in a software environment (in contrast to a hardware environment) named the Common Language Runtime (CLR). The CLR is an application virtual machine that provides services such as security, memory management, and exception handling. As such, computer code written using .NET Framework is called "managed code". FCL and CLR together constitute the .NET Framework. .NET Framework began as proprietary software, although the firm worked to standardize the software stack almost immediately, even before its first release. Despite the standardization efforts, developers, mainly those in the free and open-source software communities, expressed their unease with the selected terms and the prospects of any free and open-source implementation, especially regarding software patents. Since then, Microsoft has changed .NET development to more closely follow a contemporary model of a community-developed software project, including issuing an update to its patent promising to address the concerns. .NET Framework includes an implementation of the CLI foundational Standard Libraries. The .NET Framework Class Library (FCL) is organized in a hierarchy of namespaces.

NET platforms are encouraged to implement a version of the standard library allowing them to re-use extant third-party libraries to run without new versions of them. The .NET Standard Library allows an independent evolution of the library and app model layers within the .NET

**2.3.10**



Figure 11: Logo of MYSQL

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. A relational database organizes data into one or more data tables in which data may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyser. MySQL is offered under two different editions: the open-source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.

Major features as available in MySQL: -

* A broad subset of ANSI SQL 99, as well as extensions
* Cross-platform support
* Stored procedures, using a procedural language that closely adheres to SQL/PSM
* Triggers
* Cursors
* Updatable views
* Online Data Definition Language (DDL) when using the InnoDB Storage Engine.
* Information schema
* Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
* A set of SQL Mode options to control runtime behavior, including a strict mode to better adhere to SQL standards.

**2.3.11 DialogFlow ES**



Figure 12: Logo of Dialogflow

Dialogflow is a natural language understanding platform that makes it easy to design and integrate a conversational user interface into your mobile app, web application, device, bot, interactive voice response system, and so on. Using Dialogflow, you can provide new and engaging ways for users to interact with your product.

Dialogflow can analyse multiple types of input from your customers, including text or audio inputs (like from a phone or voice recording). It can also respond to your customers in a couple of ways, either through text or with synthetic speech.

* **Agents**

A Dialogflow agent is a virtual agent that handles concurrent conversations with your end-users. It is a natural language understanding module that understands the nuances of human language. Dialogflow translates end-user text or audio during a conversation to structured data that your apps and services can understand. You design and build a Dialogflow agent to handle the types of conversations required for your system.

* **Intents**

An intent categorizes an end-user's intention for one conversation turn. For each agent, you define many intents, where your combined intents can handle a complete conversation. When an end-user writes or says something, referred to as an end-user expression, Dialogflow matches the end-user expression to the best intent in your agent. Matching an intent is also known as intent classification.

* **Training phrases**

These are example phrases for what end-users might say. When an end-user expression resembles one of these phrases, Dialogflow matches the intent. You don't have to define every possible example, because Dialogflow's built-in machine learning expands on your list with other, similar phrases.

* **Action**

You can define an action for each intent. When an intent is matched, Dialogflow provides the action to your system, and you can use the action to trigger certain actions defined in your system.

Parameters: When an intent is matched at runtime, Dialogflow provides the extracted values from the end-user expression as parameters. Each parameter has a type, called the entity type, which dictates exactly how the data is extracted. Unlike raw end-user input, parameters are structured data that can easily be used to perform some logic or generate responses.

* **Responses**

You define text, speech, or visual responses to return to the end-user. These may provide the end-user with answers, ask the end-user for more information, or terminate the conversation.

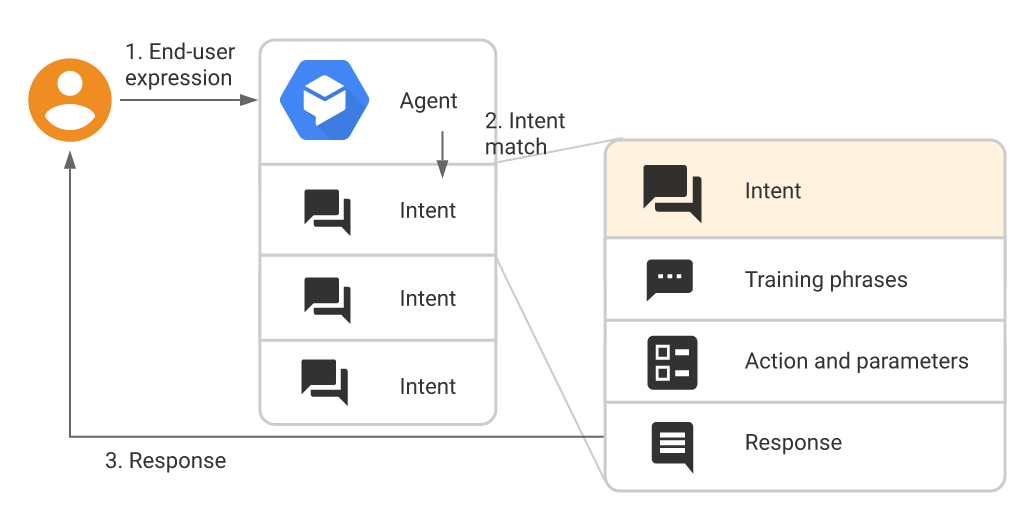


Figure 13: basic flow for intent matching and responding to the end-user

* **Entities**

Each intent parameter has a type, called the entity type, which dictates exactly how data from an end-user expression is extracted.

Dialogflow provides predefined system entities that can match many common types of data. For example, there are system entities for matching dates, times, colors, email addresses, and so on. You can also create your own custom entities for matching custom data. For example, you could define a vegetable entity that can match the types of vegetables available for purchase with a grocery store agent.

* **Contexts**

Dialogflow contexts are similar to natural language context. If a person says to you "they are orange", you need context in order to understand what "they" is referring to. Similarly, for Dialogflow to handle an end-user expression like that, it needs to be provided with context in order to correctly match an intent.

Using contexts, you can control the flow of a conversation. You can configure contexts for an intent by setting input and output contexts, which are identified by string names. When an intent is matched, any configured output contexts for that intent become active. While any contexts are active, Dialogflow is more likely to match intents that are configured with input contexts that correspond to the currently active contexts.

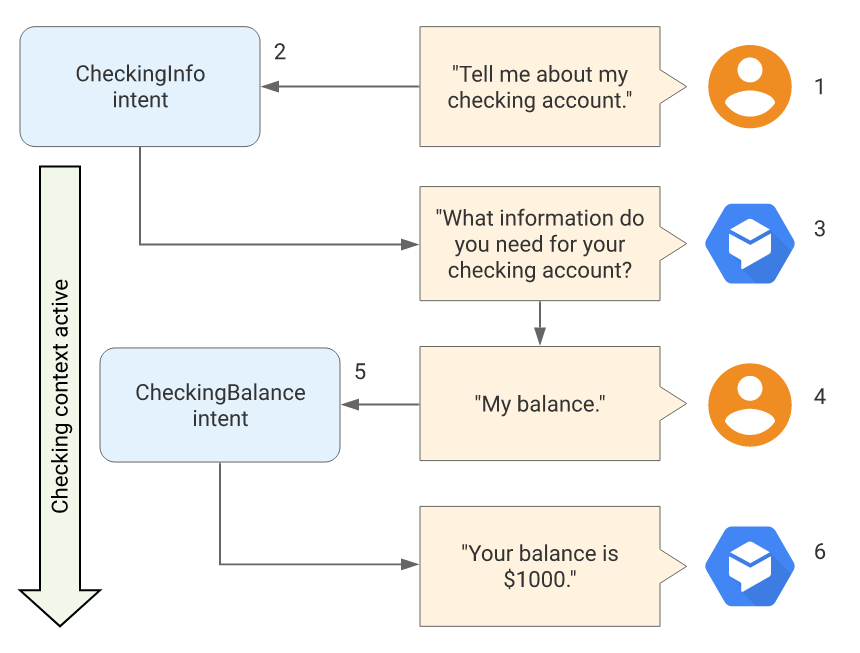


Figure 14: Example that uses context

* **User interactions with integrations**

Dialogflow integrates with many popular conversation platforms like Google Assistant, Slack, and Facebook Messenger. If you want to build an agent for one of these platforms, you should use one of the many integration’s options. Direct end-user interactions are handled for you, so you can focus on building your agent. Each integration handles end-user interactions in a platform-specific way, so see the documentation for your integration platform for details.

* **Fulfillment for integrations**

Each intent has a setting to enable fulfillment. If an intent requires some action by your system or a dynamic response, you should enable fulfillment for the intent. If an intent without fulfillment enabled is matched, Dialogflow uses the static response you defined for the intent. When an intent with fulfilment enabled is matched, Dialogflow sends a request to your webhook service with information about the matched intent. Your system can perform any required actions and respond to Dialogflow with information for how to proceed. When fulfillment is enabled, the static response you defined for the intent is only used if your webhook service fails. The following diagram shows the processing flow for fulfillment.

**2.3.12 Visual Studio Code**



Figure 15: Logo of Visual Studio Code

Visual Studio Code is a source code editor developed by Microsoft for Windows, Linux and MacOS. It includes support for debugging, embedded Git control, syntax highlighting, intelligent code completion, snippets, and code refactoring. It is free and open-source, although the official download is under a proprietary license. Visual Studio Code can be extended via [extensions](https://en.wikipedia.org/wiki/Plug-in_(computing)), available through a central repository. This includes additions to the editor and language support. A notable feature is the ability to create extensions that add support for new [languages](https://en.wikipedia.org/wiki/Programming_language), [themes](https://en.wikipedia.org/wiki/Theme_(computing)), and [debuggers](https://en.wikipedia.org/wiki/Debugger), perform [static code analysis](https://en.wikipedia.org/wiki/Static_code_analysis), and add [code linters](https://en.wikipedia.org/wiki/Lint_(software)) using the [Language Server Protocol](https://en.wikipedia.org/wiki/Language_Server_Protocol).

Visual Studio Code includes multiple extensions for [FTP](https://en.wikipedia.org/wiki/FTP), allowing the software to be used as a free alternative for web development. Code can be synced between the editor and the server, without downloading any extra software. Visual Studio Code allows users to set the [code page](https://en.wikipedia.org/wiki/Code_page) in which the active document is saved, the [newline](https://en.wikipedia.org/wiki/Newline) character, and the programming language of the active document. This allows it to be used on any platform, in any locale, and for any given programming language.

**2.3.13 Microsoft Azure**



Figure 16: Logo of Microsoft Azure

Microsoft Azure, often referred to as Azure is a cloud computing service operated by Microsoft for application management via Microsoft-managed data centres. It provides software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS) and supports many different programming languages, tools, and frameworks, including both Microsoft-specific and third-party software and systems. Azure Web Sites allows developers to build sites using ASP.NET, PHP, Node.js, Java, or Python and can be deployed using FTP, Git, Mercurial, Team Foundation Server or uploaded through the user portal. This feature was announced in preview form in June 2012 at the Meet Microsoft Azure event. Customers can create websites in PHP, ASP.NET, Node.js, or Python, or select from several open source applications from a gallery to deploy. This comprises one aspect of the platform as a service (PaaS) offering for the Microsoft Azure Platform.

* Azure Active Directory is used to synchronize on-premises directories and enable SSO
* Azure Active Directory B2C allows the use of consumer identity and access management in the cloud.
* Azure Active Directory Domain Services is used to join Azure virtual machines to a domain without domain controllers.
* Azure information protection can be used to protect sensitive information.
* Storage Services provides REST and SDK APIs for storing and accessing data on the cloud.
* Table Service lets programs store structured text in partitioned collections of entities that are accessed by partition key and primary key. Azure Table Service is a NoSQL non-relational database.

**2.3.14 Postman**



Figure 17: Logo of Postman

Postman helps in a type of software testing that involves testing application programming interfaces (APIs) directly and as part of integration testing to determine if they meet expectations for functionality, reliability, performance, and security. Since APIs lack a GUI, API testing is performed at the message layer. API testing is now considered critical for automating testing because APIs now serve as the primary interface to application logic and because GUI tests are difficult to maintain with the short release cycles and frequent changes commonly used with Agile software development and DevOps.

**2.3.15 GCP**



Figure 18: Logo of Google Cloud Platform

Google Cloud Platform (GCP), offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products, such as Google Search, Gmail, file storage, and YouTube. Alongside a set of management tools, it provides a series of modular cloud services including computing, data storage, data analytics and machine learning.it provides a series of modular cloud services including computing, data storage, data analytics and machine learning. Registration requires a credit card or bank account details. Google Cloud Platform is available in 25 regions and 77 zones. A region is a specific geographical location where users can deploy cloud resources. Each region is an independent geographic area that consists of zones. A zone is a deployment area for Google Cloud Platform resources within a region. Zones should be considered a single failure domain within a region.

**Chapter- 3**

**Objectives of the project**

Today, the Internet have created a new economic scenario that not only stresses on the classical concept of the **“Customer”** but also on the modern concept of **“Service”.** It is this level of service that dictates whether a customer get an Ease or not.

To provide a high accessibility of service we design the “**ChatBot**” so that potential customers need not go to a customer care for personal information and service .

***There are several objective of this “ChatBot” are following given below:-***

* This gives all the information about the booking to provide better service for the customer and it understands consumer preferences.
* This ChatBot enables you to take your booking and other information without so much help of customer care support.
* It provides the facility to the customers who want to everything online due to lack of time.
* It provides facility to payment online by the credit card, Debit Card etc.
* It’s providing the full detail about the booking and related information about flight tickets.
* This “ChatBot “enables you to “Save Time” .
* It provides better security and good Informative service to the customer about his booking and other operation.

**3.1 SYSTEM ANALYSIS**

System Analysis is a problem-solving activity that requires intensive communication between the system users and system developers.

It is basically a process of gathering information about the “ChatBot” for continuous improvement in the Application.

Systems analysis is the process of observing systems for troubleshooting or development purposes. The project is totally built at administrative end and thus only the administrator is guaranteed the access.

The purpose of the project is to build a “ChatBot” to reduce the manual work for customer care executive and provide the ease to the customer.

“ChatBot” tries to enhance access to care and improve the continuity and efficiency of service to the customer. It is easy to maintain, ready to run, scalable, affordable and reliable cost saving tool.

* 1. **EXISTING SYSTEM**

In the existing system everything is done manually. There is no central server. It works in distributed environment.

* It is less user-friendly.
* Customer must go to customer care for every piece of information.
* It is difficult to resolve the problem of every customer on the same time.
* It is a time-consuming process.
* It is also not accessible at every time.
  1. **PROPOSED SYSTEM**

The aim of proposed system is to automate more things in flight booking service. The proposed system can overcome all the limitations of existing “ChatBot”.

* In the proposed system “ChatBot” is improved in such manner that they can automate the every flight booking and its related service.
* customer don’t have to wait for the customer care executive to see their personal information.
* customer can take advantage of self service at 24\*7 which is a game changing part of application.
* It minimizes the time needed for manual data entry and customer support.

**Chapter- 4**

**FEASIBILITY STUDY**

The feasibility study investigates the problem and the information needs of the stakeholders. It seeks to determine the resources required to provide an information systems solution, the cost and benefits of such a solution, and the feasibility of such a solution. The analyst conducting the study gathers information using a variety of methods, the most popular of which are:

* Interviewing users, employees, managers, and customers.
* Developing and administering questionnaires to interested stakeholders, such as potential users of the information system.
* Observing or monitoring users of the current system to determine their needs as well as their satisfaction and dissatisfaction with the current system.
* Collecting, examining, and analyzing documents, reports, layouts, procedures, manuals, and any other documentation relating to the operations of the current system.
* Modelling, observing, and simulating the work activities of the current system.

The goal of the feasibility study is to consider alternative information systems solutions, evaluate their feasibility, and propose the alternative most suitable to the organization. The feasibility of a proposed solution is evaluated in terms of its components. These components are:

**4.1 Economic feasibility**

The economic viability of the proposed system. The proposed project's cost and benefits are evaluated. Tangible costs include fixed and variable costs, while tangible benefits include cost savings, increased revenue, and increased profit. A project is approved only if it covers its cost in a given period of time. However, a project may be approved only on its intangible benefits such as those relating to government regulations, the image of the organization, or similar considerations.

**4.2 Technical feasibility**

The possibility that the organization has or can procure the necessary resources. This is demonstrated if the needed hardware and software are available in the marketplace or can be developed by the time of implementation.

**4.3 Operational feasibility**

The ability, desire, and willingness of the stakeholders to use, support, and operate the proposed computer information system. The stakeholders include management, employees, customers, and suppliers. The stakeholders are interested in systems that are easy to operate, make few, if any, errors, produce the desired information, and fall within the objectives of the organization.

**Chapter- 5**

**DESIGN STRATEGY**

**5.1 Hardware Requirement**

* Processor: Pentium Processor ISA 32 Family
* Secondary Storage: 80 GB HDD
* RAM: 52X CD ROM Drive
* Memory: 4 GB RAM
* Network Adapter: Ethernet Adapter
* Modem: 128kbps Voice Fax Data
* Others: 17" Colour Monitor, Printer, Scanner, Keyboard, Mouse.
* Server to Deploy the Application

**5.2 Software Requirement**

* Platform: Windows 10
* Operating System: Windows XP Professional (64 GB)
* Platform: Visual Studio code
* Front-End Tool: HTML, CSS, JavaScript, React.
* Back- End Tool: Node.js, Typescript, C#.
* Scripting Tool: Java script, XML (style Sheet (.xsl))

**Chapter - 6**

**DATA FLOW DIAGRAM**

A data flow diagram (DFD) is a graphical representation of the “flow” of the data through an information system modelling its process aspects.

A Data Flow Diagram (DFD) is a graphical representation of the “flow” of data through an information system. A data flow diagram can also be used for the visualization of the data processing (structured design).

**6.1 Elements of a DFD**

There are four key elements in a Data Flow Diagram; Processes, Data Flows, Data Stores & External entities and each element is drawn differently.

**6.2 Process entity**

The Process entity identifies a process taking place; it must have at least one input and output. A process with no input is known as a “miracle process” and one with no output is a “black hole process”. Each process has the following:

* A Number
* A Name
* A Description
* At least one input
* At least one output

**6.3 Data flow entity**

* The Data Flow Entity identifies the flow of data between processes, data stores & external entities. A data flow cannot connect an external entity to the data source; at least one connection must be with a process. There are also physical flows, i.e., those that use a physical medium. Each data flow has the following:
* A Name (Noun)
* A description
* One or more connections to a process

**6.4 Data Store entity**

The Data Store entity identifies store of data, both manual and electronic. Each data store has the following:

* A Number
* A Name
* A Description
* One or more input data flows
* One or more output data flows.

**6.5 External entity**

The External entity identifies external entities which interacts with the system, usually clients but can be within the same organization. Each external entity has the following:

* A Name (Noun)
* A Description

**6.6 SYMBOLS USED IN DFD**

* **SQUARE:** It defines the destination of the system. It shows the input/output used in the system.
* **CIRCLES or BUBBLES*:*** It represents a process that transfers incoming data flow into outgoing data flow
* **ARROW:** It indicates data flow-data in motion. It is a pipeline through which information flows.
* **OPEN RECTANGLES**: It is used to represent file/Database.

**6.7 Level 0 Dfd**

A context diagram is a Top Level (also known as “zero level”) Data Flow Diagram. It only contains one process node (“Process 0”) that generalises the function of the entire system in relationship to external entities.

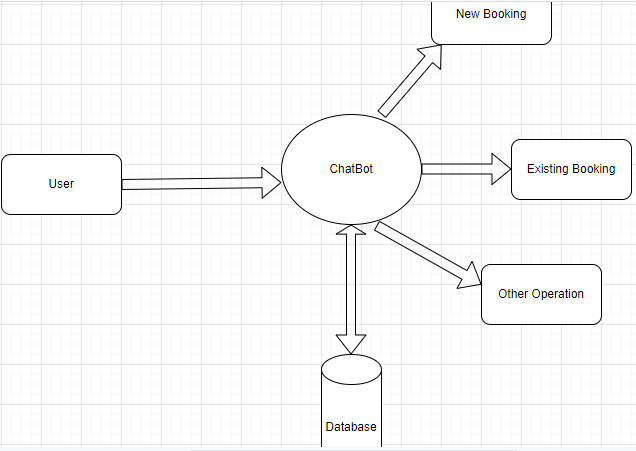


Figure 19: Level 0 Dfd

**6.8 Level 1 Dfd**

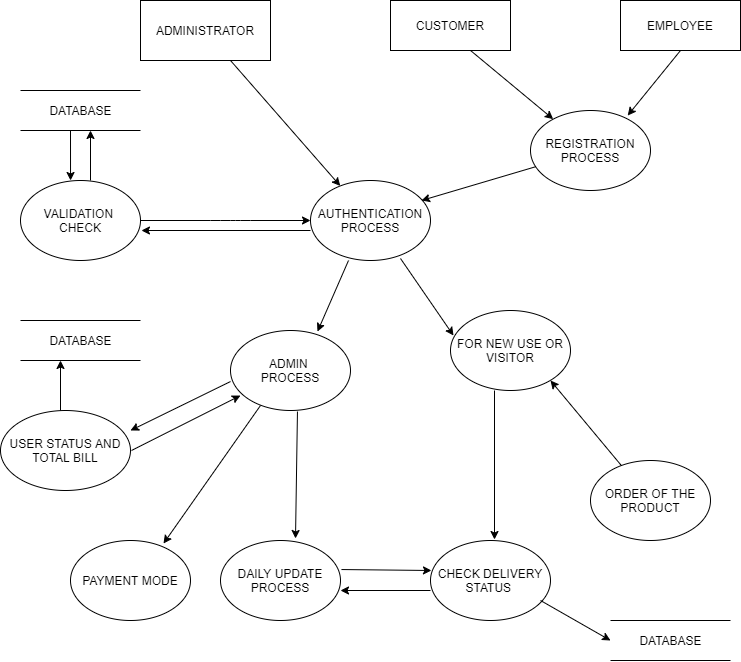


Figure 20: Level 1 Dfd

Note: - this is only for example

**6.9 Level 2 Dfd**

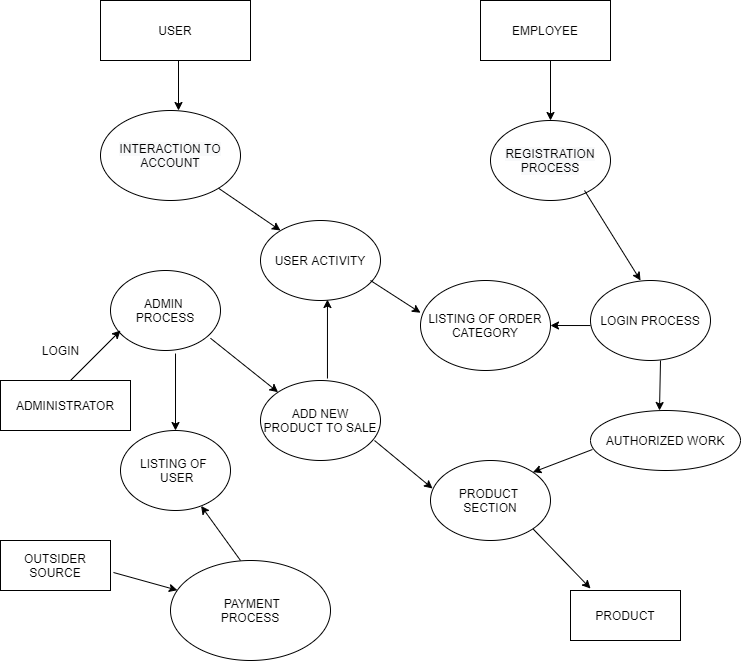


Figure 21: Level 2 Dfd

Note:- this is only for Example

**6.10 E – R DIAGRAM**

An Entity Relationship (ER) Diagram is a flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system.

ER Diagram are often used to design or debug Relational Database in the fields of Software Engineering, Business Information System, Education and Research. Also known as ER Models, they use a defined set of symbols such as rectangles, diamonds, oval, connecting lines to show the Entity, Relationships, Attributes.

* ENTITY: - these are real world object that participates in constitution of the system.
* RELATIONSHIP: - An association of several entities is called relationship.
* ATTRIBUTES: - The data items which describes the features of entity is known as attributes.

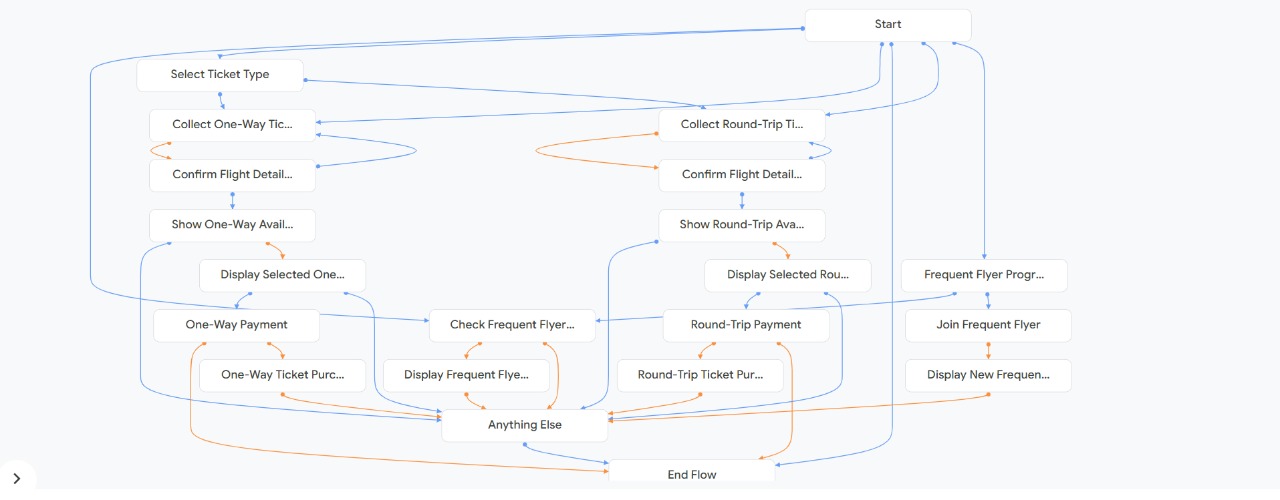


Figure 22: E-R Diagram

**Note: - Due to company policy it is only for example**

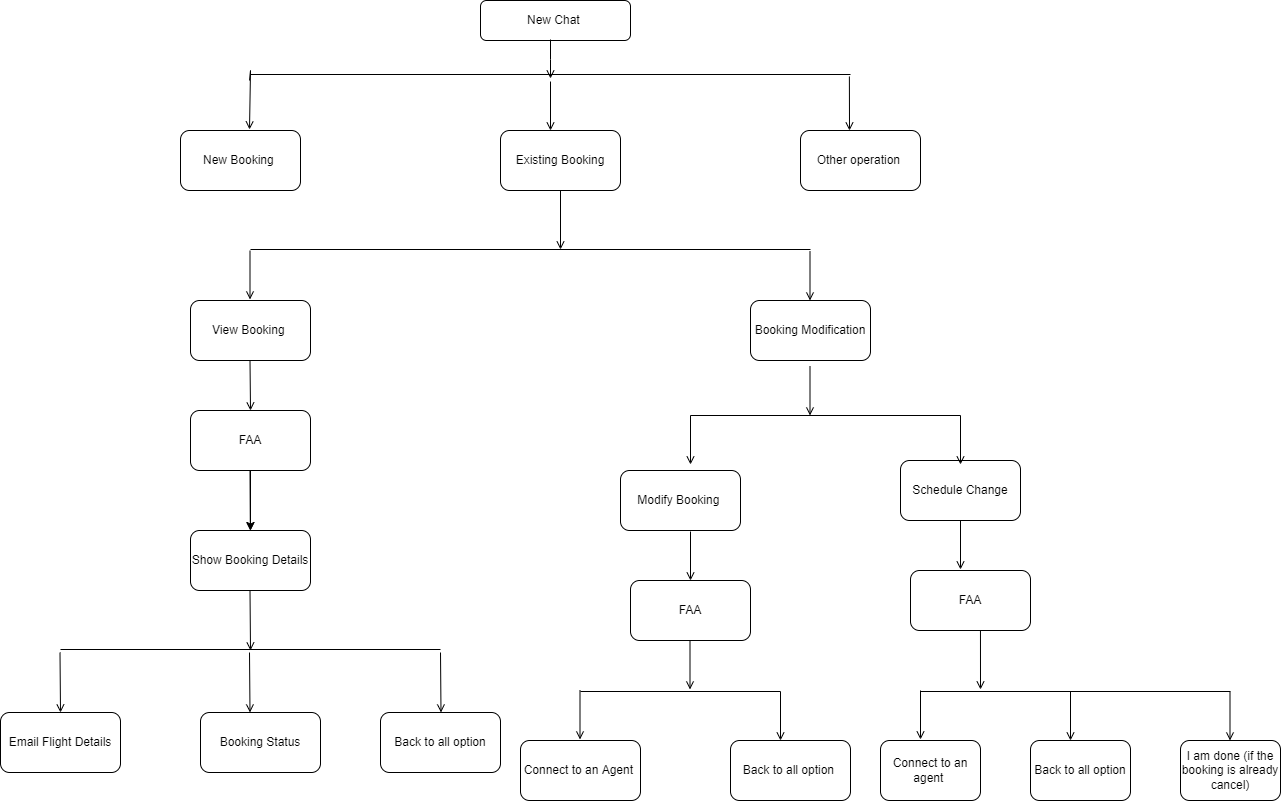


Figure 23: Flow Diagram

**Chapter - 7**

**MODULE DESCRIPTION**

The ‘Module Description’ of the Project is as follows:

Modularization is a very important part of the project development. Since, the software development is a combination of several different types of tasks, which cannot be done at a time. So, to do the work we have taken modules. The process of modularization has been followed in the preparation of this ‘Mini Project’. We know that it is quite difficult to work on the whole “ChatBot” at a time.

So, it is the task of the programmer to break/split the “development of ChatBot” into various small modules so that one can handle them effectively and it is less time consuming if we develop different modules in parallel.

We will try to split it in modules, so that there will be no inter-dependency among them.

This project developed for “ChatBot” is broken into following modules.

* Admin Module (ChatBot API)
* User Module (ChatBot UI)
* Customer Service Module (Nice as a Service)
* NLP Engine (Google Dialogflow)
* **ADMIN MODULE: -** Admin module is mainly for the owner of the system. This module performs the work of regular checking process. So, this module is responsible for updating the information continuously if any, in the system so that one can find timely information. But the main function of this module is to handle security of this system. Different types of authentications and authorization process is performed for this purpose.
* **USER PAGE MODULE: -** User page module is for front page of our application. It is the main page of the system in which all the links of this system are kept. Now the main thing is how to make the home page interactive and user friendly so that there should not be any problem for the user. This is the task of the system administrator to make the home page Interactive.
* **CUSTOMER SERVICE MODULE: -** In the customer service module, we use Nice to interact with our customer. We Also use DialogFlow ES in our project for NLP .

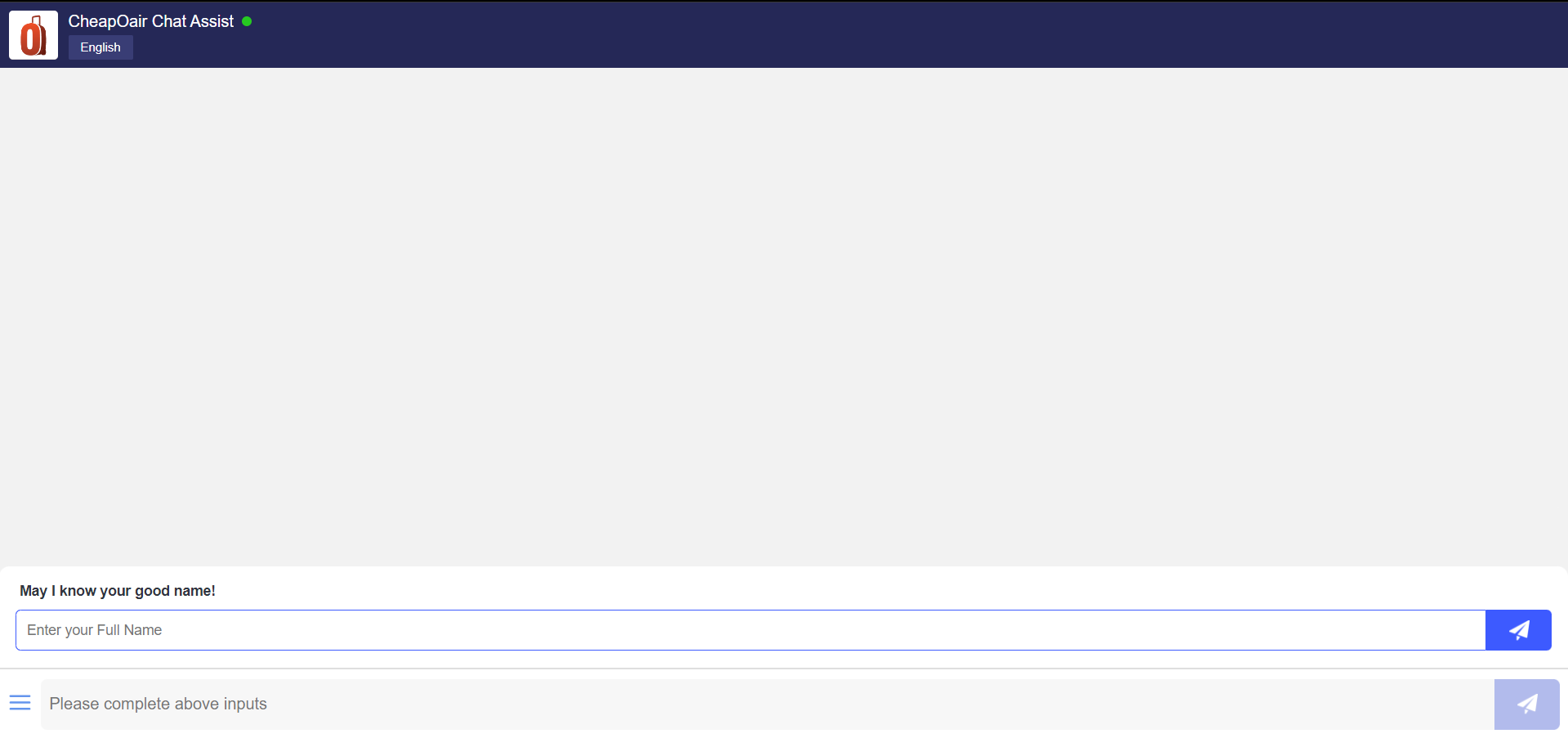


Figure 24: Landing page of Chatbot

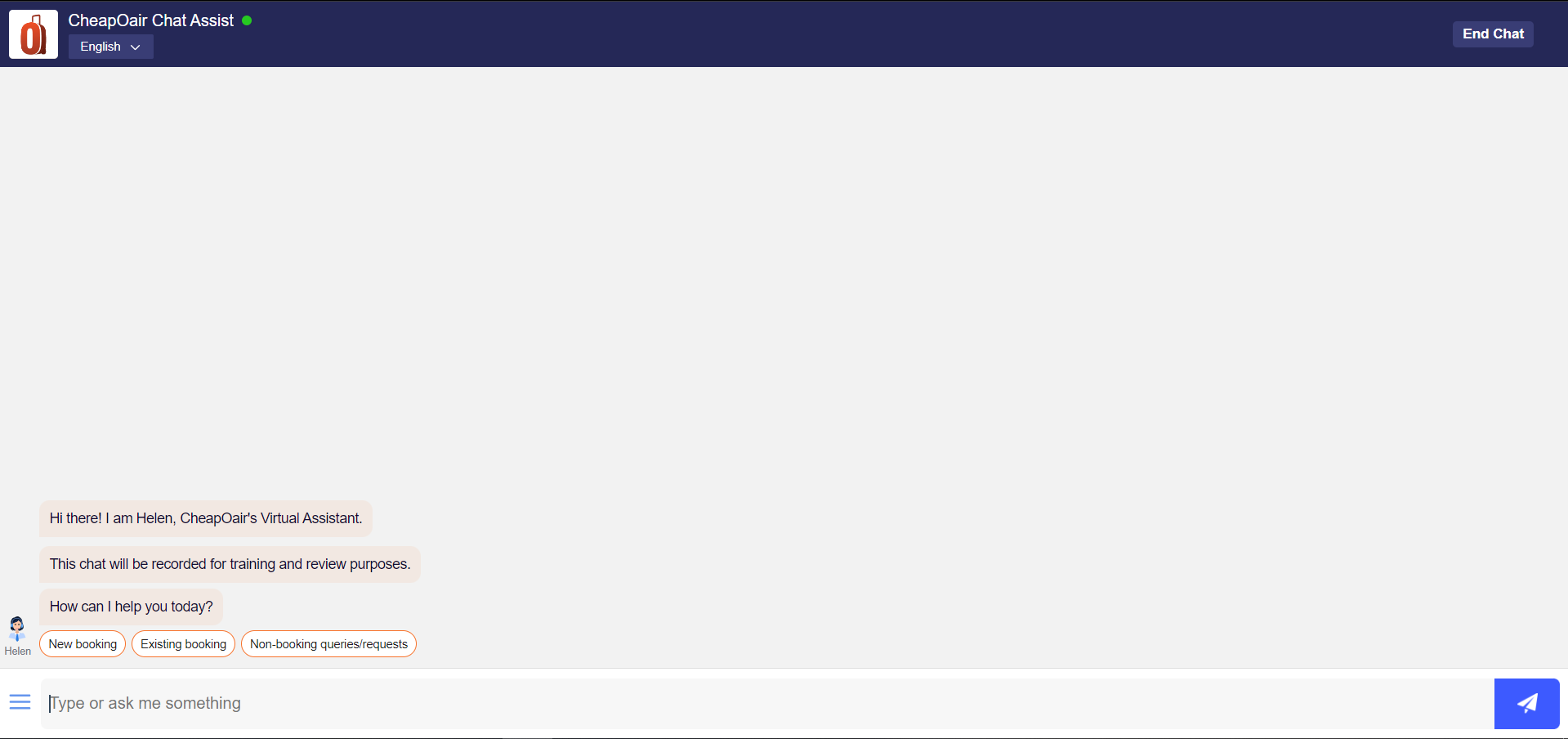


Figure 25: Initial message from Chatbot

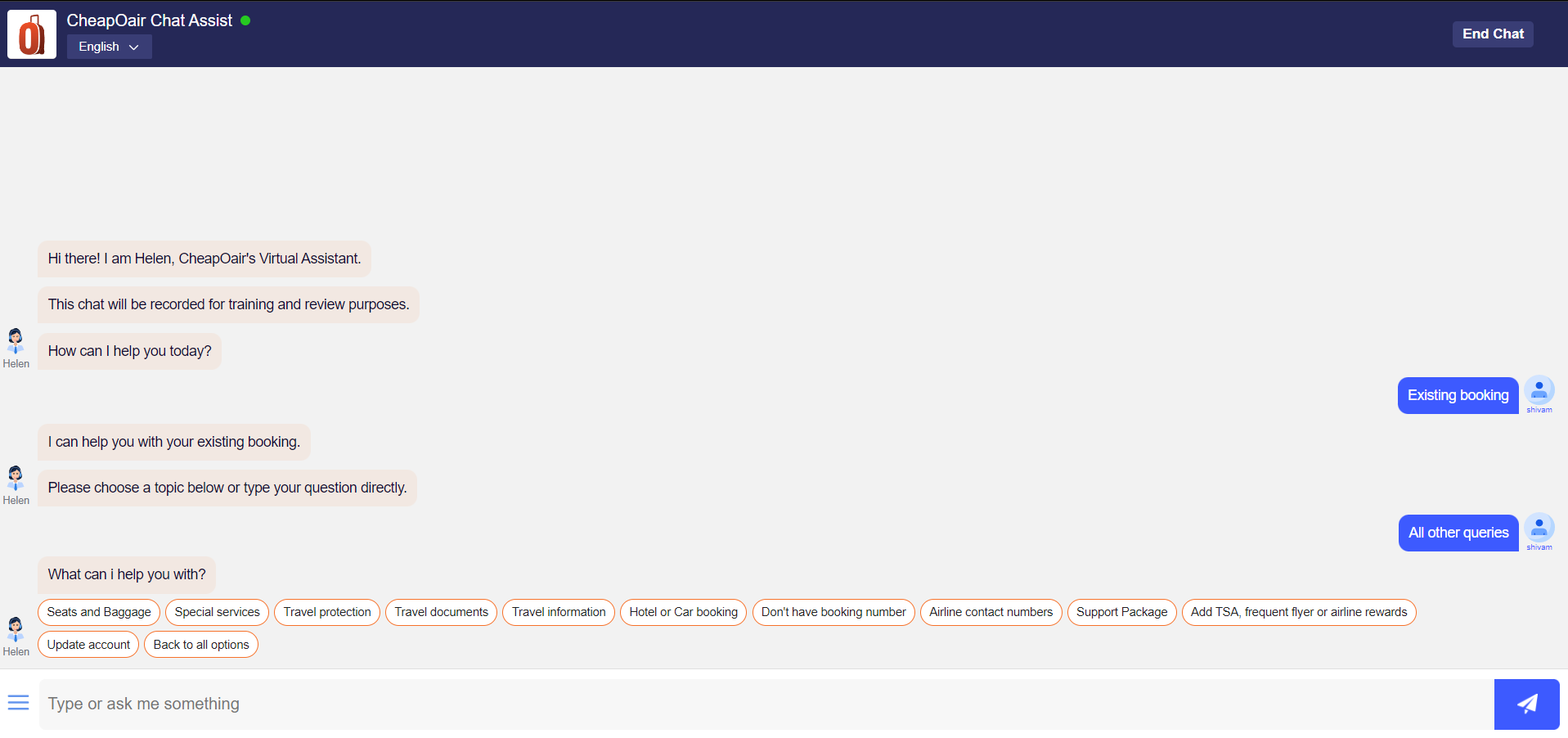


Figure 26: Chatbot Interaction Page

**Chapter - 8**

**8.1 Conclusion**

“ChatBot” is basically a web-based application, that helps the user/customer to book a ticket and get information about booking with the help of “Chatbot”. It keeps a vast number of products that the user will want to purchase. It allows the user to do many operations related to flight ticket booking.

It can store the information of Flight Booking, description, Ticket details, transaction details, showing the details of any other Information etc. The Evaluation of “ChatBot” is measured by containment rate of chat which is not transfer to the Agents. Now our containment is something around 45% but we want to increase it up to 70%. The Evaluation of “ChatBot” is also measured in the term of revenue saved and generated through it. With the help of “ChatBot” we want to automate the things under the umbrella of Flight tickets booking and operation related to it.

**8.2 References**

* [www.Zepmeta.co.in](http://www.Zepmeta.co.in)
* <https://cloud.google.com/dialogflow>
* <https://cloud.google.com/dialogflow/docs>
* <https://cloud.google.com/dialogflow/cx/docs>
* [www.javatpoint.com](http://www.javatpoint.com)
* [www.tutorialspoint.com](http://www.tutorialspoint.com)
* [www.w3schools.com](http://www.w3schools.com)
* <https://reactjs.org/>