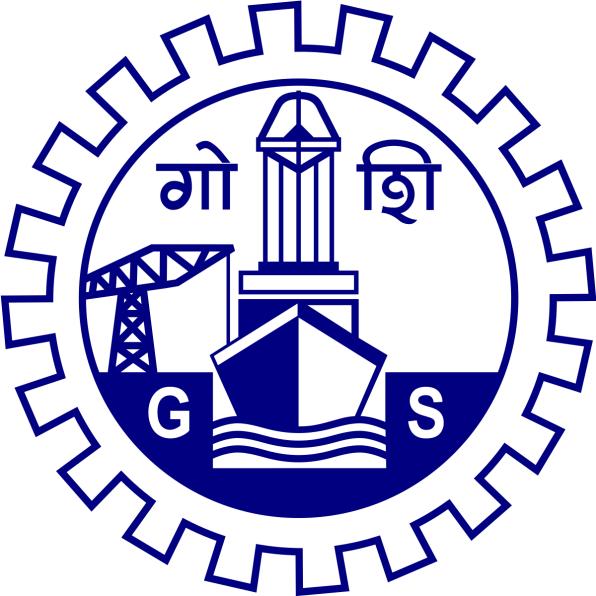
**INTERNSHIP REPORT**

AT

GOA SHIPYARD LIMITED

VASCO-DA-GAMA

20-09-2021 – 30-10-2021

****

**SUBMITTED BY:**

SAISH PAWAR,

DON BOSCO COLLEGE OF ENGINEERING,

FATORDA-GOA

DEPT: COMPUTER ENGINEERING

**DECLARATION**

I, Saish Pawar, currently pursuing B.E. in Computer Engineering, third year from Don Bosco College of Engineering, Fatorda – Goa and hereby declare that this report has been submitted by me on fulfillment of one month internship training at Goa Shipyard Limited, Vasco from 20-09-2021 to 30-10-2021.

Place: Vasco-Da-Gama

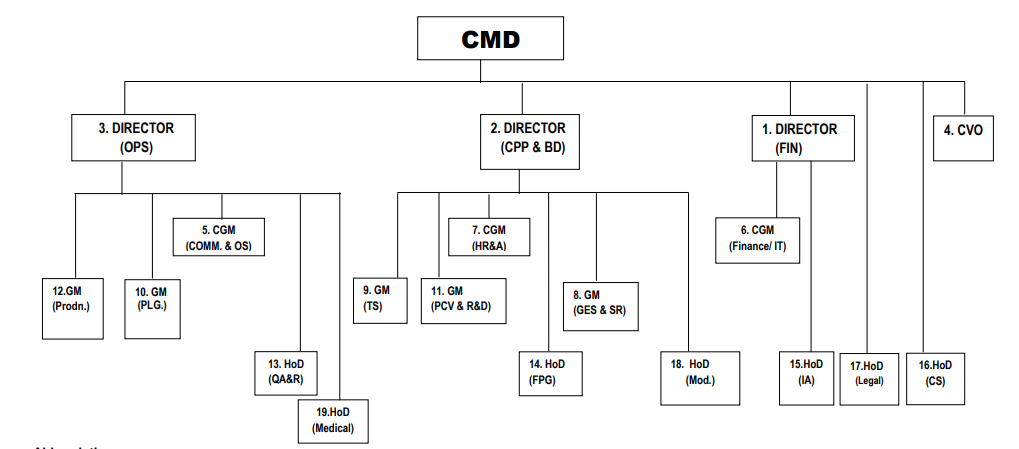
Date: 30-10-2021

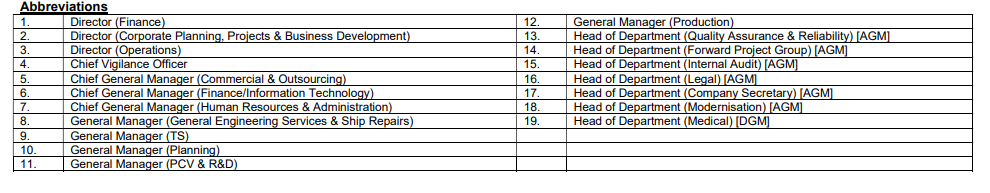
**ACKNOWLEDGEMENT**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **SRNO** | **TOPICS** | **PAGE NO.** |
| 1. | Introduction | 1 |
| 2. | Organization Chart | 3 |
| 3. | Vision and Mission | 4 |
| 4. | Facilities at GSL | 5 |
| 5. | Project Overview | 6 |
| 6. | Technologies & Frameworks Used | 7 |
| 7. | Database Structure | 16 |
| 8. | Data Dictionary | 21 |
| 9. | Wireframes | 22 |
| 10. | Color Palettes | 24 |
| 11. | The Website | 25 |
| 12. | The Chatbot | 28 |
| 13. | Implementing Features | 32 |
| 14. | Difficulties Faced | 34 |
| 15. | Conclusion | 35 |

* **Introduction**
* Goa Shipyard Limited (GSL) established in 1957, is a leading ISO 9001-2015 certified shipyard on the West Coast of India, functioning under the administrative control of Ministry of Defence, Govt. of India.
* GSL is strategically located on the banks of river Zuari in Goa, a major international tourist destination well connected by its international airport and major port enroute all important shipping lines.
* Beginning as a small barge building yard, GSL has garnered reputation as one of the most sophisticated ship builders in the Country. For over four decades, GSL has designed, built and commissioned a wide range of sophisticated vessels for varied applications in the defence and commercial sectors with special expertise in building modern patrol vessels of Steel and Aluminium hull structure.
* GSL is equipped with an in-house design capability and carries out its own Research & Development, in the process of developing an in-house product range that efficiently meets the specific requirements of clients in the defence as well as commercial sectors for the years.
* To date GSL has built different vessels like barges, tugs, landing craft, Missile craft, offshore patrol vessels, AOPV’s, survey vessel and other vessels for the Indian Navy and Coast Guard and for export to countries like Yemen, Sri Lanka, Mauritius etc.
* The Yard’s infrastructure is capable of docking 120M vessels upto 6000T and a Ship Transfer Area of around 13600 sq. mts and thus it can now count itself as India’s first Defence Shipyard to be equipped with a modern Ship lift facility for launching and docking of ships. The commissioning of the ship lift has given impetus to repair activities.
* At present GSL has commenced the production of guided missile frigates (02 nos) in collaboration with Russia which has an indigenous make BrahMos missile, SONAR system, ACTAS system etc.
* **Organization Chart**

****

****

* **Vision And Mission**
* **Facilities at GSL**
* **Project Overview:**
* I was assigned a project to work on a “ChatBot” for the official GSL Website.
* The aim of the ChatBot is that it should be able to solve any basic questions/queries a user might have when they visit the website.

**Approach:**

* Creating a mock website to test the bot.
* A user can either choose to minimize the window, which transforms it into a button, or continue the conversation with the bot.
* Maintaining a knowledge base database, so that whenever a user asks a query outside of what the bot is trained for, it can be added to the knowledge base, so that in future the bot can be trained to answer the same question.
* Storing all basic questions and answers in a knowledge database
* Storing department details in department\_details database
* Storing navigation links in navigation database
* Manually integrating answers for specific use-cases such as Company information or Contact Details.
* The bot greets the user according to the current local time whenever he/she visits the website.
* The basic working around this is that, when a user replies, the bot searches through the database to find a similar question & replies the answer. If it finds that the question is not found; It first replies the user with a “Sorry, couldn’t understand your query” message and then adds the query to another database, where it can be trained to answer the same question in the future.

**Technologies & Frameworks Used**

* **HTML5 :**

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 and scripting languages such as JavaScript.

* I used this to build the basic structure for the bot and the mock-website.
* **CSS :**



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

* The main Design and Styling of the ChatBot + Mock Website was achieved using CSS ( Cascading Style Sheets).
* **Bootstrap:**

 Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

* I used this framework to quickly develop a mock-website to test the chatbot. Also, this framework was very useful in making critical components of the chatbot UI. Some examples of use-cases include: Navigation menu, footers, chatbot window, input-fields, Buttons, etc as well as website responsiveness.
* **JavaScript:**



JavaScript, often abbreviated as JS, is a programming language that conforms to the ECMAScript specification. JavaScript is high-level, often just-in-time compiled, and multi-paradigm. It has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

* JS was a integral part of the project. Most of the scripts and functioning of the ChatBot was possible due to JavaScript.
* **jQuery:**

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript. As of May 2019, jQuery is used by 73% of the 10 million most popular websites.

* **Animations** for the Chatbot was majorly coded using jQuery.
* Many functions of the bot were implemented in jQuery.
* **AJAX:**

Asynchronous JavaScript and XML, while not a technology in itself, is a term coined in 2005 by Jesse James Garrett, that describes a "new" approach to using a number of existing technologies together, including HTML or XHTML, CSS, JavaScript, DOM, XML, XSLT, and most importantly the XMLHttpRequest object.

When these technologies are combined in the Ajax model, web applications are able to make quick, incremental updates to the user interface without reloading the entire browser page. This makes the application faster and more responsive to user actions.

* Used this technique, to send POST requests to get data from the database. AJAX helped in connecting the website with the web server.
* **PHP:**

The PHP Hypertext Preprocessor (PHP) is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web based software applications. PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites. It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.

* I used PHP to manage the MySQL Database; executing queries on the database and extracting information.
* It was also used to connect the chatbot/website to the MySQL Database.
* **MySQL:**

 MySQL is currently the most popular database management system software used for managing the relational database. It is open-source database software, which is supported by Oracle Company. It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database. It is commonly used in conjunction with PHP scripts for creating powerful and dynamic server-side or web-based enterprise applications. MySQL follows the working of Client-Server Architecture. This model is designed for the end-users called **clients** to access the resources from a central computer known as a **server** using network services. Here, the clients make requests through a graphical user interface (GUI), and the server will give the desired output as soon as the instructions are matched. The process of MySQL environment is the same as the client-server model.

* **MySQL** was the main database framework I used for this project. The syntax for queries like Updating, Inserting, Deleting, Creating Tables is very easy to understand and implement. It is also a very fast and quick method, to create a Database.
* Various Datatables were also created as per requirements.
* **XAMPP:**

XAMPP is an abbreviation where X stands for Cross-Platform, A stands for Apache, M stands for MYSQL, and the Ps stand for PHP and Perl, respectively. It is an open-source package of web solutions that includes Apache distribution for many servers and command-line executable along with modules such as Apache server, MariaDB, PHP, and Perl.

XAMPP helps a local host or server to test its website and clients via computers and laptops before releasing it to the main server. It is a platform that furnishes a suitable environment to test and verify the working of projects based on Apache, Perl, MySQL database, and PHP through the system of the host itself. Among these technologies, Perl is a programming language used for web development, PHP is a backend scripting language, and MariaDB is the most vividly used database developed by MySQL.

* The testing and working of the project required a webserver. XAMPP was the perfect solution for hosting the Chatbot on a local Webserver .
* **Git & Github:**

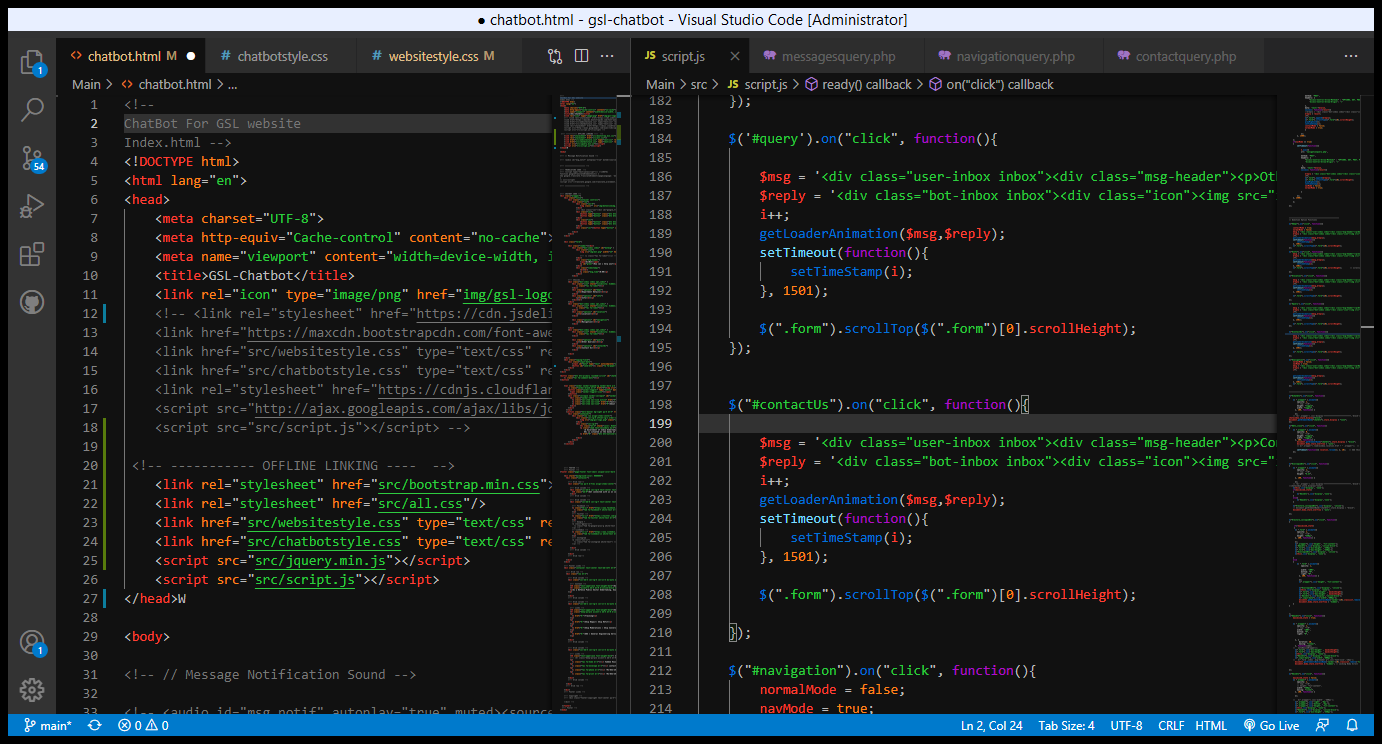
An important aspect of the project was **maintaining** it. Creating backups of day-to-day work updates and storing it, is easily solved by using **Github** and the **Git** framework. Git is a free and open distributed version control system, a tool to manage your source code history.

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.

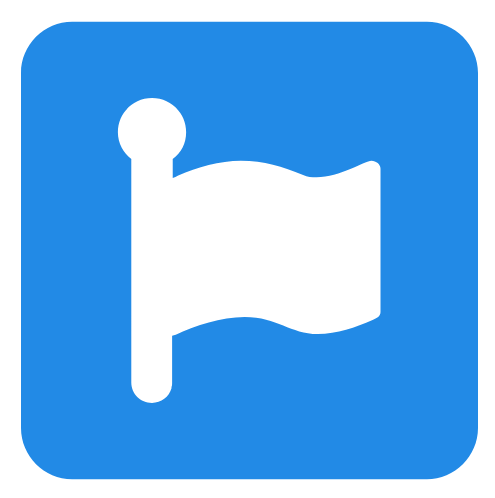
In short, Git is the tool, and GitHub is the service for projects that use Git.

* To maintain the project, I created a git repository on GitHub - <https://github.com/sashpawar11/gsl-chatbot>
* The versioning of the project was automated using git through the IDE – Visual Studio Code.
* **Integrated development environment (IDE):**
  + Visual Studio Code ( Vscode )

Visual Studio Code is an integrated development environment made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

****Snapshot of the GSL-ChatBot project source code in Vscode:

* **Font Awesome:**



Font Awesome is a font and icon toolkit based on CSS and Less. As of 2020, Font Awesome was used by 38% of sites that use third-party font scripts, placing Font Awesome in second place after Google Fonts.

* All the icons required for buttons, and other utilities were implemented using this library.
* **Database Structure**
* The main database for the chatbot was termed as “gslchatbot”.

MySQL Query:

CREATE DATABASE gslchatbot;

**DATATABLES:**

* The project required creating datatables for the functioning of different features such as Department Details, answering basic User messages, NewQueries etc.

1. **knowledge\_base**
   * This datatable stored all the basic questions and answers which the bot referenced to, whenever a user replied to the bot.
   * It contains columns – id\*(PrimaryKey), questions, answers.
   * All sample questions and its answers were stored in this datatable using the MySQL Insert Queries.

MySQL Query:

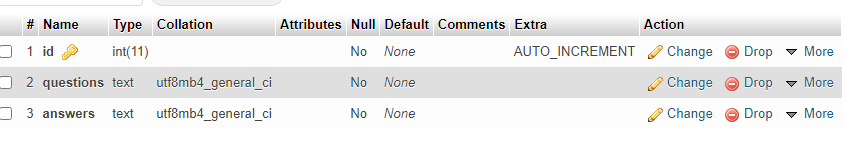
CREATE TABLE knowledge\_base(

id INT AUTO\_INCREMENT PRIMARY KEY,

questions TEXT NOT NULL,

answers TEXT NOT NULL

);

****

1. **department\_details**
   * This datatable was created to store the contact details of the departments in GSL.
   * It was referenced by the bot, when a user wanted contact details about any specific department.
   * Columns – opt\_id\*(PrimaryKey), dept\_name, dept\_email, dept\_phone.

MySQL Query:

CREATE TABLE department\_details(

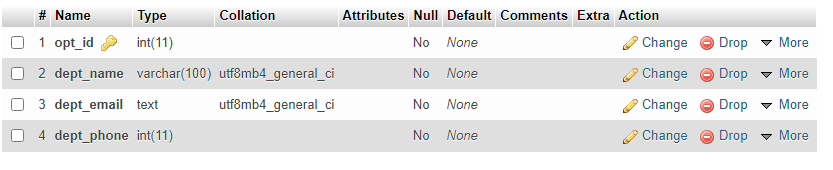
opt\_id INT NOT NULL PRIMARY KEY,

dept\_name VARCHAR(100) NOT NULL,

dept\_email TEXT NOT NULL,

dept\_phone INT NOT NULL

);

****

1. **navigation**
   * This datatable stores all the links to different pages for the website for navigation-based questions.
   * It contains columns – opt\_id\*(PrimaryKey), header,link.
   * Link columns contains the hyperlinks to the different pages, which will be returned to the user in the form of clickable links.

MySQL Query:

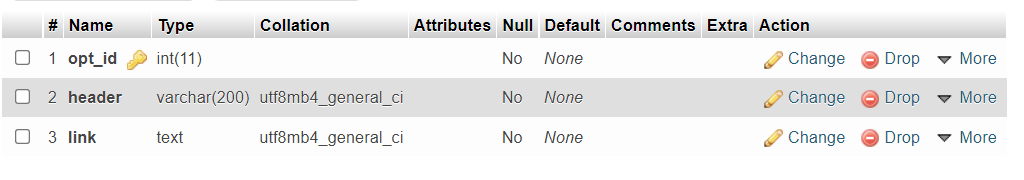
CREATE TABLE navigation(

opt\_id INT NOT NULL PRIMARY KEY,

header VARCHAR(200) NOT NULL,

link TEXT NOT NULL

);

****

1. **newqueries**
   * This datatable stores all the unknown queries the bot encounters in any conversation.
   * The same questions are later added to the knowledge base so that the bot is trained to answer it in future.
   * Columns – ID\*(PrimaryKey), que

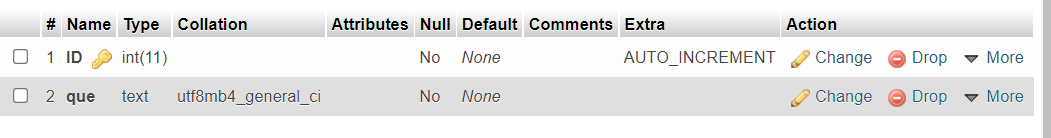
MySQL Query:

CREATE TABLE newqueries(

ID INT AUTO\_INCREMENT PRIMARY KEY,

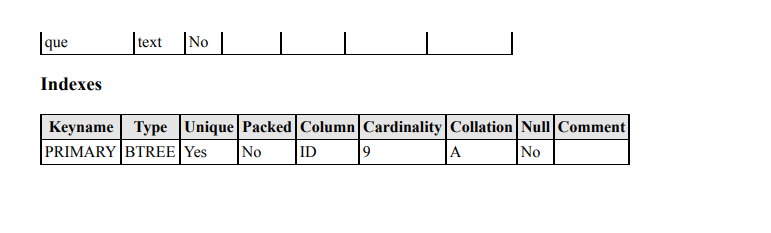
que TEXT NOT NULL

);

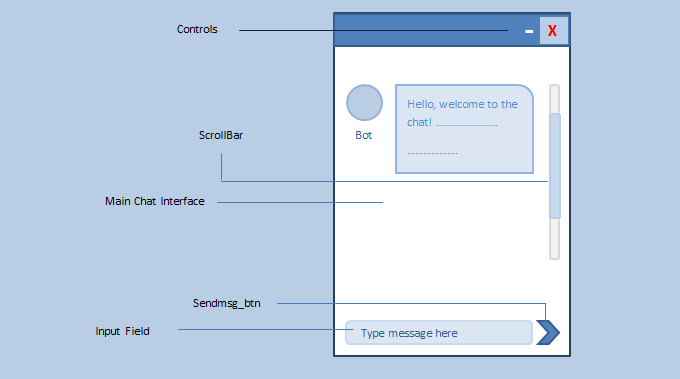
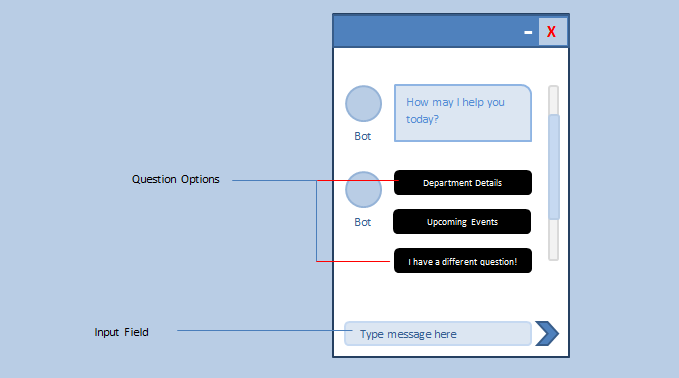
****

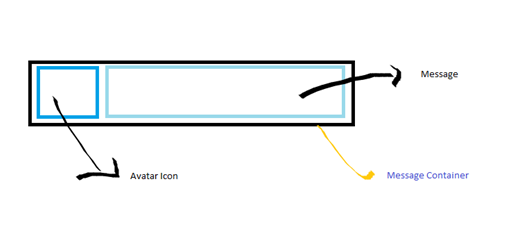
* **Data Dictionary**

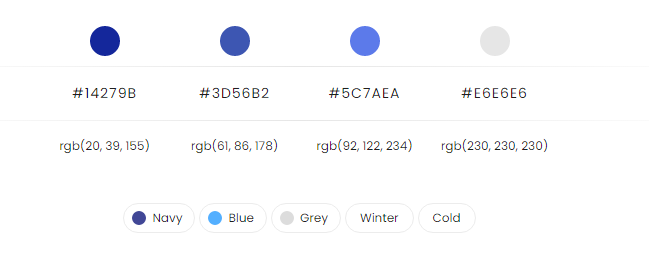
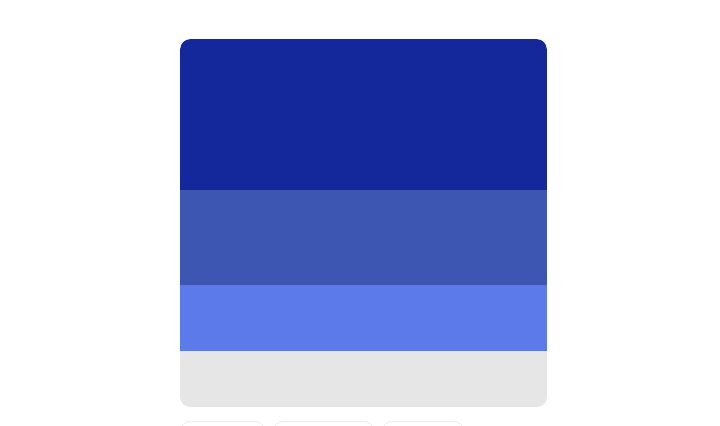


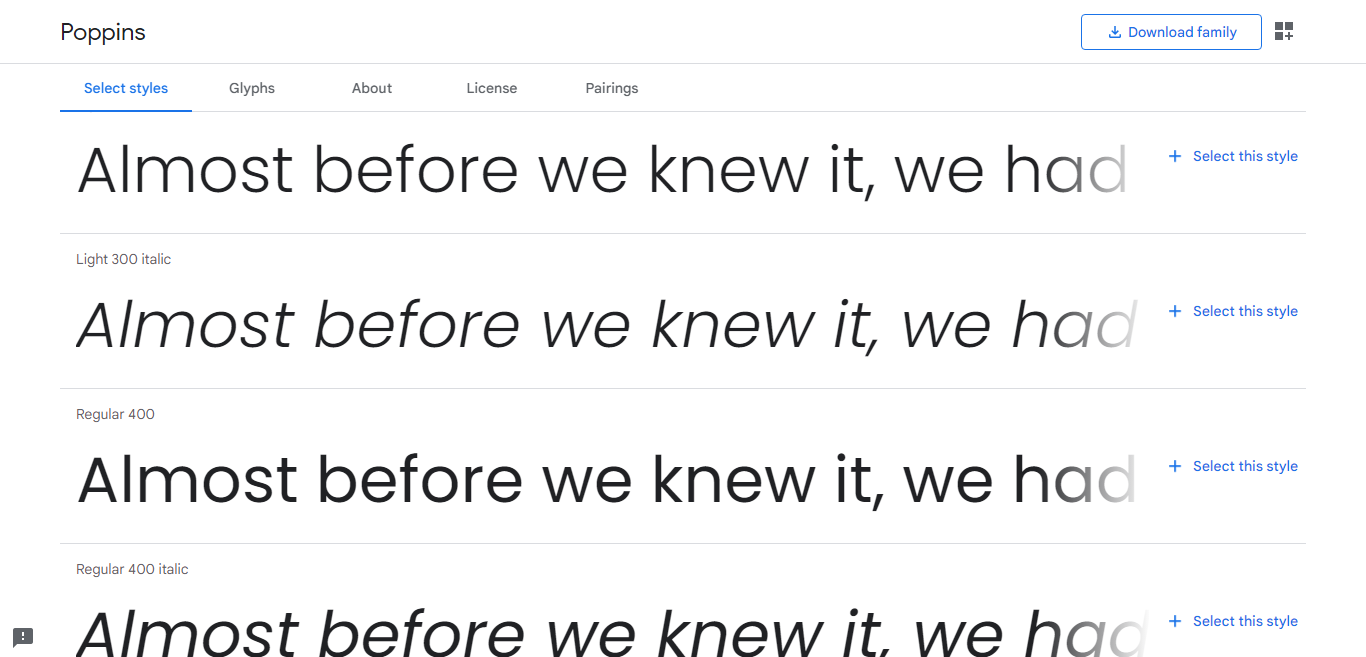
****

* **Wireframes**
* The initial process involved wire-framing the interface for the bot-window. I started designing a rough interface using MS Word.

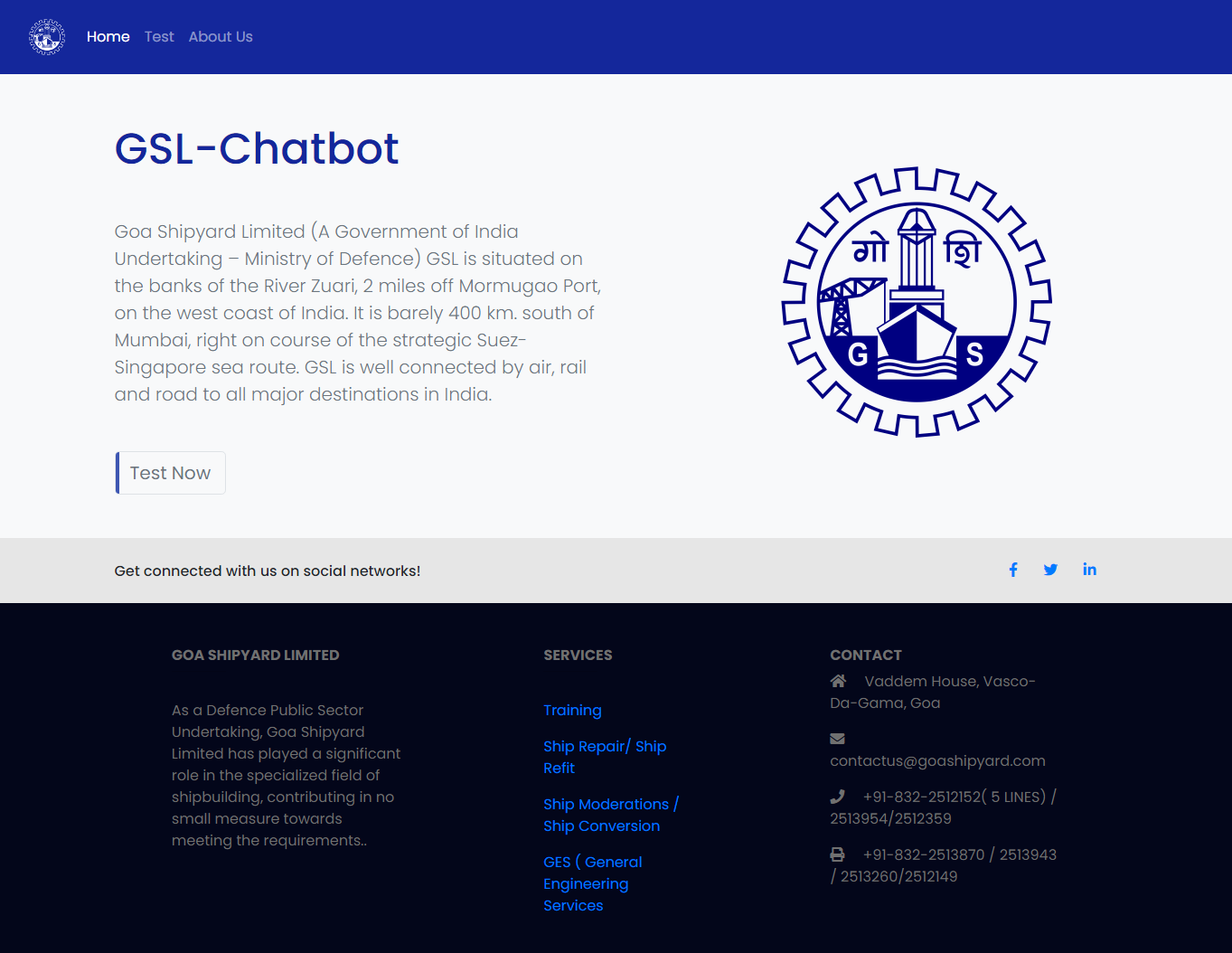




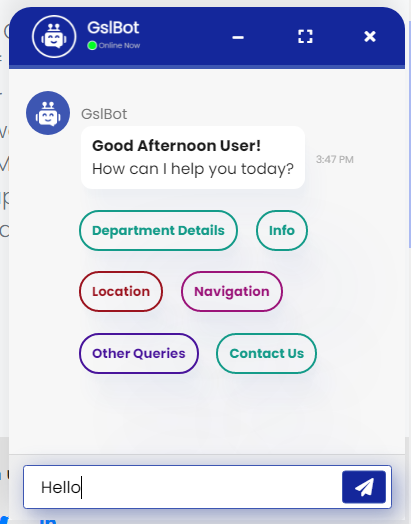
* Sketched out rough wire-frames for the Chatbot Window, Chatbot Button, Message Structure.
* **Color Palettes**
* In the design process, I chose this Color Palette for the main theme of the Chatbot + the mock website.
* **The Website**
* To test the ChatBot, it needed a mock website.
* For designing and making a website quickly, since it is just a mock website, I used the Bootstrap Framework. The variety of features it has is great. From buttons, to forms, to navigation bars.
* Bootstrap also helped to easily make the website responsive.
* For the website, I used these bootstrap components:
  + - Footer
    - Buttons
    - Forms
    - Navigation Bar
    - Wrappers
    - Grid
* For the colors, I used the same color palette which will be used for the Chatbot UI as well.
* Thus giving the website a uniform look in terms of color-theme and layout.
* Separated the styling of the website and the ChatBot in different files, so that it would be easier to integrate the bot in the official website later. 🡪 “*src/webstyle.css”*
* Added a Hero Banner, displaying the details about the company and the company Logo.
* Footer – Contains Information and Useful Links from the Company Website.
* Imported and used the font – “Poppins” for both, the website as well as the ChatBot Interface.



* Also added a Socials Contact Links Bar.

****

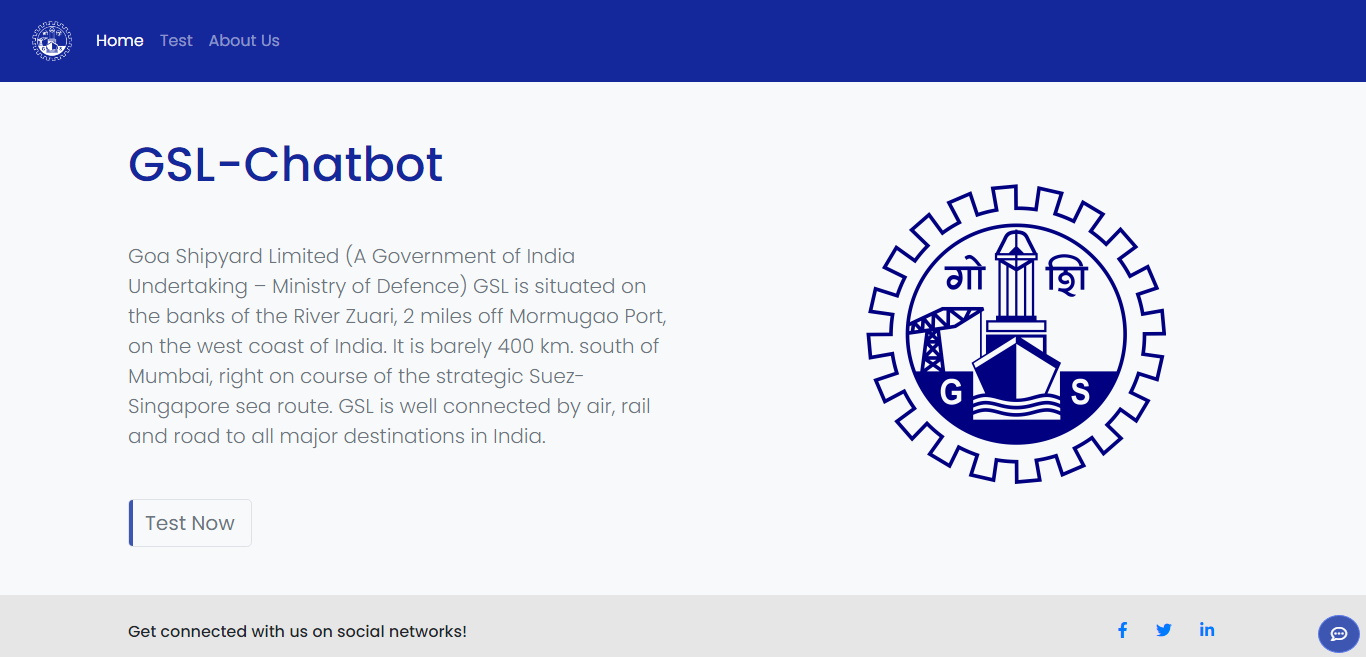
* **The ChatBot**
* The first and foremost step towards developing the bot was to make the interface window, the Chat Button, binding Click Functions, jQuery Animations and testing the UI. The later part was just, integrating it with the database and adding features one at a time.



**Fig 1.1**

* The **basic components** of the ChatBot are: (*Refer figure 1.1)*
  + - **Chat Button**
    - **Control Bar - - - - - - - - - -(1)**
      * Title
      * Collapse/Restore Buttons
      * Minimize/Maximize Buttons
      * Close Button
    - **Chat Section - - - - - - - - - (2)**
      * Loader Animations + Delay
      * Bot-Messages
      * Options
      * User-Messages
    - **Input-text Area - - - -- - - -- (3)**
    - **Send Button - - - - - - - (4)**
* **Chat Button**

This is the main point-of-access button to start a conversation with the ChatBot. The button is positioned fixed at the right-bottom corner of the website. When the user clicks it, the Chatwindow interface pops-up and the user can start the conversation. – Refer Fig 1.2 and Fig 1.1.



**Fig 1.2**

**Chat Button**

* **Control Bar - - - - - -** *( 1 > Fig 1.1)*

It is located at the top of the Chatbot Window. It Consists of the GSLBot Logo, GSLBot Title + Online Status and Control Buttons

* Collapse/Restore Buttons – These buttons are used to hide/ unhide the Chat Windows
* Maximize/Minimize Buttons – These buttons change the state of the ChatWindow – Fullscreen / Standard.
* Close Button – This button closes the chat window and reloads the page, thus the conversation with the bot are not saved.
* **Chat Section - - - - - -** *( 2 > Fig 1.1)*

It is the main section of the Chatbot Window. The Chat sections is implemented using Forms. All the messages and interactions take place in this section.

* Loader Animations + Delay – Normally, there isn’t much delay before the bot replies to any user query, thus to make it seem more natural, I added a delay before the actual bot replies appear to the user. During this delay, a loading-style animation plays for a short interval of time till the message is displayed to the user.
* Bot-Messages – The messages are structured as – 1) Bot Avatar, 2) GSLBot Header, 3) Message, 4) Timestamp
* Options – These are the highlighted options which the bot provides to the user in the beginning.
* User Messages – These are the messages which will appear after the user types any message.
* **Input Text Area - - - - - -** *( 3 > Fig 1.1)*

This is the main input-field, where a user types any message. A user can either hit the Enter key or click the Send Button to send the message.

* **Send Button - - - - - -** *( 4 > Fig 1.1)*

This button is used to send messages to the ChatBot. It takes the data from the input field and executes queries.

* **Implementing Features**
* **GREETING MESSAGE:**
  + When a user starts a conversation with the ChatBot, it first greets the user with a greeting message, wishing the user with a ***“Good Morning***” or ***“Good Afternoon”*** or ***“Good Evening”*** based on the current local time.
* **INFO:**
  + Provides a brief introduction about the company and history, along with a picture.
* **LOCATION:**
  + Returns a interactive Google map location widget to the user, showing the location of the company.
* **DEPARTMENT DETAILS:**
  + The users can get contact details of the various departments in the company via this feature. The ChatBot provides the user to select from multiple departments, queries with the details stored in the database, and returns it to the user in a formatted fashion.
* **NAVIGATION:**
  + If a user needs to visit a specific page on the website, it can ask the bot via this feature. The chatbot provides various options to the user, which they can select and visit via the link returned by the chatbot. It works in a similar manner as the department details feature, it queries the navigation database for the respective link of the webpage, and returns it to the user in the chat.
* **CONTACT US:**
  + Provides official contact details of the Company.
* **OTHER QUERIES :**
  + If the user has some other query, they can use this feature to ask general queries to the chatbot.
* Supplementing this, many more features were added to the Chatbot. Such as the Maximize/ Minimize feature, which changes the size of the chat window; Collapse / Restore feature – transforms the window into a collapsed state/ normal state; Hover effects for the options; Animations; Responsiveness to different screen resolutions; Close button transformations.
* **Difficulties Faced**
* A major hurdle was deciding which technologies to use to develop the chatbot. After researching I finally decided to use the technologies mentioned above.
* There were various errors that occurred during the development of the project. Such as, the database not being properly connected by the php code resulting in errors, the chatWindow not displaying accurately on the website etc. All these were fixed after debugging and by trial&error.
* The queries made when interacting with the bot, showed various POST request errors like ‘CORS’. This was due to the project files not being properly stored in the local server folder. The bug was fixed by moving the project files inside the XAMP server folders.
* **CONCLUSION**