A Systematic Literature Review on ChatGPT’s Impact on Software Development

Shahrin Binte Ali
Department of ICT
ID : 23549908034

Trisha Shaha
Department of ICT
ID : 23549908035

Maliha Nawar
Department of ICT
ID : 23549908045

***Abstract*—In near future, ChatGPT may play vital role in software development because it has almost every possible information (till September, 2021) about programming languages. In software development, chatGPT contributes by generating code snippets, test cases and providing any information required during software development cycle[5]. But alongside these, the model is the reason of having some defects related to frequent errors in coding, lack of creativity and complexity, out of date data, bias in training data etc.. This paper aims to focuses on a systematic literature review that explores the productive, ethical use of ChatGPT in software development. The paper has highlighted challenges, benefits, adoption factors of ChatGPT in software development[3].This review resulted in answers of some research questions about the usage, benefit, efficiency and limitation of chatGPT in software development.**

***keywords*— ChatGPT; Software development; Impact of chatGPT; Advantage of chatGPT; ChatGPT for developer.**

I. Introduction

Established by Generative Pre-trained Transformer (GPT) architecture , ChatGPT is an AI chatbot tool which was published by Open Artificial Intelligence (AI) at the end of November, 2022[7]. This is a chatbot that interacts with users using openAI’s language model. Though ChatGPT is impressive and can create results for writing songs, poetry,tales, essays, and other things, it has some limitations[12]. If users ask the bot questions, it replies with relevant, convincing and fitting replies[9].

Besides answering questions and text generation, ChatGPT can also produce code snippets. So ChatGPT can be utilized by developers for testing and debugging codes[9]. It is feasible to automate routine coding and to streamline the process[4]. Developer will be greatly benefited with the usage of chatGPT. Although there are some notable ways on how ChatGPT can impact the developers, chatGPT has certain drawbacks too for developers such as frequent errors in coding, lack of creativity and complexity, out of date data, bias in training data etc[16]. The objective of this paper is to do a systematic reporting on impacts of ChatGPT on software development. How ChatGPT could play a key role in the development of software in coming days, what are the challenges and opprtunities that we are going to face. Among all the articles that are published on chatGPT and it’s impact on software development, the focus has been narrowed on the benefits and factors of chatGPT that need to be adopted.

In this study, Systematic Literature Review or SLR methodology has been adopted to do this study.A systematic review is a systematic and explicit method in which first problems are defined, then evaluated and after evaluating some related research questions are generated.Collected data related to the research is analyzed to answer the questions.

To perform SLR method in this paper a protocol has been followed that involved step by step procedure. To fulfill the motive 9 research questions were generated which fall into 4 different categories- Usage, Benefit, Efficiency, Limitation.In usage, the study includes finding out how chatGPT is used in developing software or in software generally.In benefit, the target was to find out what are the advantages of chatGPT in software development field. In efficiency, the motive was to figure out how chatGPT is effective for software developers.And in limitation, its drawbacks are discussed.After answering research questions, results and findings were discussed along with their significance.Selected article’s provided data were assigned to attribute set to answer research questions.We also discussed the future impacts of those review results.

In this paper, sections are designed as follows: in Methodology, review protocol and research questions were developed.Studies or works related to this paper have been discussed in Related works section, Review results are discussed and analyzed the answer extracted from research questions in Opportunities and Challenges section.In Adoption factor, the future scope of research has been explored depending on the review results.In Discussion, main findings, limitation and future scopes are discussed.Lastly the report concluded in conclusion section.

II. Related Works

Bale et al.[4] focused on developments in software engineering utilizing chatGPT. Considering the challenges chatGPT is currently facing such as compromise of sensitive information, data bias and leakage of private information, their study aimed to improve software development process using chatGPT in their paper.
ChatGPT has been applied in software development tasks such as code generation, solving programming bugs, programming Numerical methods, code completion and in practicing for computer science exam. Beganovic et al. [5] performed a systematic literature review on the application of ChatGPT in software development. Their results and the offering for the field were examined. The findings of the review offered remarkable contributions to the understanding and

future direction of using ChatGPT in the sector of software development.

ChatGPT's contribution in the development of software or in software engineering initiated the argument whether chatGPT is better than human or not in the field of software development. Nascimento et al.[15] examined a comprehensive comparison between software engineers and AI-based solutions(ChatGPT) considering various evaluation criteria such as the reliability of AI-based methods, promoting human-machine collaboration and understanding task suitability for humans or AI. An empirical investigation was conducted, differentiating the performance of software engineers and ChatGPT across different evaluation metrics. For the empirical study, a ChatGPT-generated code versus code produced by developers were uploaded in Leetcode which is an online platform for coding interview.

Bug fixing is identifying and repairing errors to improve the program's functionality and overall user experience. Developers need to deal with bug fixing process during the development of software. Sobania et al. [17] evaluated ChatGPT on the standard bug fixing benchmark set, QuixBugs, and compared the performance with the results of different other methods reported in the study. This feature of chatGPT support software developers in finding and fixing software bugs and increase the system's bug fixing performance. Surameery et al.[18] emphasized on the characteristics of ChatGPT and how chatGPT can be utilized in assisting debugging process, bug prediction and bug explanation to help solve programming problems. ChatGPT's strength can be integrated with other debugging tools. This will be more effective in identifying and resolving bugs[5].

Using chatGPT new software architect can be built. Ahmed et al.[2] used a case study that concerns collaboration between chatGPT and a novice software architect to create a service-based software. ChatGPT is the emerging solution that can collaborate Software Development Bots (DevBots) trained on large language models, with architects'intelligence and artificially intelligent decision support to enable a human-bot ACSE.

Kashefi et al [14] observed the capability of chatGPT in differentiating codes written by humans and machines. The possibilities of ChatGPT in developing software was examined by making ChatGPT assess in code making for numerical algorithms in various programming languages, improving user's written codes, finishing missing parts of the numerical codes, rewriting present codes in other languages, parallelizing codes which are serial.

ChatGPT is able to assist developers in making test cases. Like, creating test inputs and writing test cases covering contrasted code paths and edge cases. Jalil et al.[11] highlighted on the potential effects of ChatGPT on software testing education, where chatGPT was tasked with answering common questions in a popular software testing curriculum. ChatGPT's performance was assessed in a software testing scenario. ChatGPT gave right answers or partially correct answers in 55.6 percent of cases, correct or partially correct explanations in 53 percent of cases.

The study of Fraiwan et al.[7] explored chatGPT's impact on various fields, including software engineering, education, healthcare, and marketing. Possible applications of advanced language Chatbots (ChatGPT) and it's research direction, drawbacks in each of these fields were discussed. In the study, it was observed that ChatGPT's application in software engineering has the potential to improve efficiency and usefulness in the software development procedure, refine the quality of software documentation and enhance collaboration.

III. METHODOLOGY

Systematic review methodology is the method that gives results by analyzing all the facts. After attempting some research questions and answering the questions there has been an endeavour to provide perception on chatGPT's impact on software development. There are several process in SLR . Attempting one by one, finally the study reached to discussion on open areas. According to fig 1,at first a review objective is defined where some research questions regarding the topic is generated.Figure 1 shows the details of SLR which are described as follows[16]

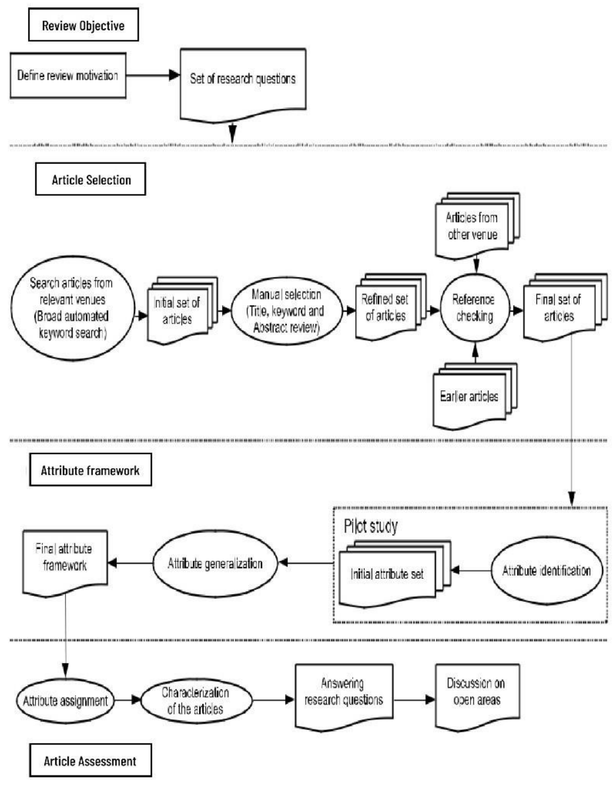


Fig. 1. Overview of systematic literature review

A. Research Questions

Generated research questions defined in Table I discusses on the aspects of Usage, Benefit, Efficiency, Limitation of chat-GPT in software development.These research questions falls into chatGPT and software development related researches that provides information of this research's utility.

B. Article Selection

Article selection process is described in figure 2 where articles are included through automated key search followed by manual selection of the articles.Then final set of articles are got through reference checking.

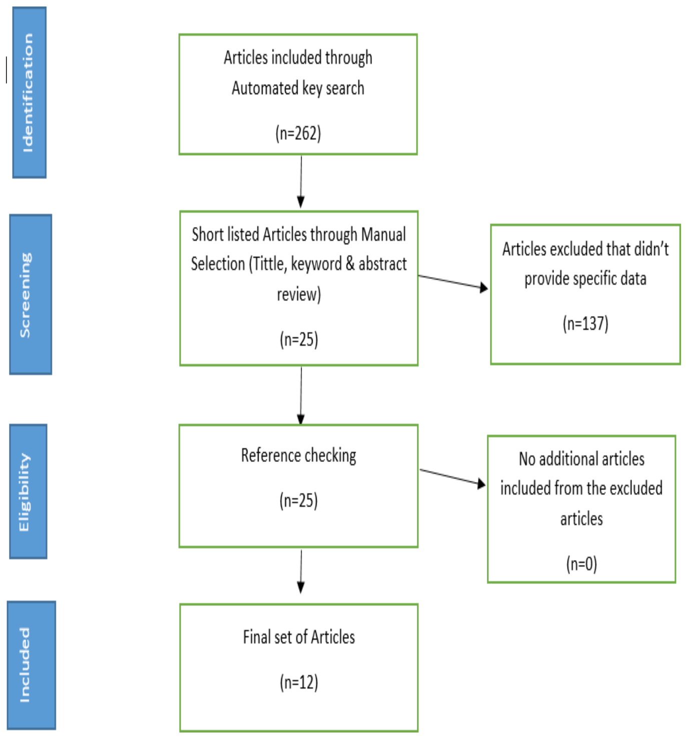

Identification
Screening
Eligibility
Included
Articles included through Automated key search (n=262)
Short listed Articles through Manual Selection (Tittle, keyword & abstract review) (n=25)
Articles excluded that didn't provide specific data (n=137)
Reference checking (n=25)
No additional articles included from the excluded articles (n=0)
Final set of Articles (n=12)

Fig. 2. Prisma Diagram for Article Selection Procdure

Inclusion criteria To define the generated research questions, selected articles were selected in order to follow some criteria[16].Keeping in mind that the selected articles must have those criteria

- Articles were related to chatGPT based researches and specially focused on the application of chatGPT in software development.Subject criteria for those articles were limited in the research for chatGPT's impact on software development.
- Articles must be related to chatGPT and software development.Articles that published research regarding chat-GPT's effect on software development were considered only.
- Selected articles were mostly published online, conference and in journal.

Utility of the articles that falls in the above mentioned criteria were explored by analyzing the title,keyword and abstract.

Automated keyword search In SLR methodology, article selection procedure starts by automated keyword search.This search system is done by searching through keywords related to the research topic from digital library.Most of the articles were searched from website of various digital library such as researchgate,Ieee computer society, google search engine and google scholers.These are the platforms where chatGPT based researches are mostly published.Through tittle, keyword and abstract, automated keywords for selecting articles were found. All the articles were published from year 2021 to 2023.

This search began by defining the search within inclusion criteria.Then the search went to the digital library with keywords related to research topic.Some search terms were used as follows:

for ”ChatGPT” search terms; ”chatGPT”, ”artificial intelligence”, ”openAI”,”chatbot using AI tool” were used

for ”Software development” search terms; ”software development”,”automating tests”,”QA” were used.

for ”ChatGPT's impact on software development” search terms; ”chatGPT 's impact on software development”,”chatGPT for developers” were used.

After automated keyword search, it ended up with more than 200 articles. Among which more than 150 were conference papers and journals others were some related websites and blogs.

Manual selection In automated keyword search, it is not easy to find specific articles that will focus on the impact of chatGPT on software development.To narrow down the related search by only focusing on the articles that provides specific information, the study came up with another procedure of short-listed articles[16];manual selection.After going through title,keyword and abstract, it was manually selected some 25 articles that provided necessary data and helped in answering the research questions.Articles that could not provide specific data regarding the research were excluded after manual selection.Also number of articles were reduced in order to make it effective.

Reference checking A non recursive search were performed to make sure if any relevant articles went missing after manual selection.But no additional articles appeared after doing that[16].

Final set of articles Finally, 12 articles which are journal papers and 5 websites with their published year mentioned in reference were selected.

C. Attribute Framework

In SLR method next step is attribute framework defined in figure 1 (c) where attribute were generalized after a pilot study of identification and initial setting of attributes which is described bellow

Attribute Identification Attributes were set on the basis of two category (1) related to research questions (2) area of the review.Based on these two criteria, the study went through a

TABLE I
RESEARCH QUESTIONS

Category	***Research questions***	***Main motive***
Usage	How ChatGPT helps in generating code snippets for specific problems	To highlight the basic and important usage of chatGPT for developers
Benefit	How ChatGPT can assist with automating unit tests	To ensure the advantage of chatGPT in automating unit test
	How can ChatGPT play important role in Automating QA	To ensure what role chatGPT plays in QA
Efficiency	How ChatGPT can help generating test cases based on parameters	To check how efficient chatGPT is in generating test cases based on parameters
	How ChatGPT analyses codes and propose best security methods	To ensure it's efficiency in code analyzing
Limitation	How effective ChatGPT is in detecting frequent errors	To find out if chatGPT is effective in detecting frequent errors in code
	How ChatGPT can provide critical thinking and complexity	To find out if chatGPT can provide critical thinking
	How capable ChatGPT is to tackle unique problems How will ChatGPT perform without out of date data	To find out if chatGPT can solve unique problem To check if chatGPT can perform without data

TABLE II
ATTRIBUTE FRAMEWORK

Attribute	***Sub Attribute***	***Brief Description***
Case study	research papers related to chatGPT's impact on software development	what are the advantages of chatGPT in software development, usage of chatGPT for developers
Study target	impacts of chatGPT in software development	benefit and drawback of chatGPT in software development
Study type		case study, designing and implementation of system, comparative
Data source	chatGPT, reseach paper related to chatGPT and software development	Online journals and articles,newspaper
Methodology	Method, impacts	method used in the study, designing the system, research method
Results	Usage, advantage and limitation of chatGPT	Why chatGPT is effective for developers and what problem it causes in software development
Evaluation		Evaluation on chatGPT's impact on software development

pilot study that includes some steps in figure 1.
15 articles(from 25 articles) were studied and by observing their constructions, 8 attributes were finalized that defined those articles and research questions.Those attributes are described in Table II

After that for an attribute sub-attributes were formed by exploring same type of articles that had some similar words relevant to the attribute(e.g. ”chatGPT for developers”, ”chatGPT in software development”, for study target attribute).A brief description for each attribute/sub-attributes were provided also.

Attribute generalization and final attribute framework Lastly , in attribute framework, some sub-attributes are generalized to simplify for multiple number of sub-attributes. For example, in ”Case study” attribute initially sub-attributes were ”chatGPT” and ”software development” but later it was generalized as ”research papers related to chatGPT's impact on software development”. For ”Study target” attribute, ”chatGPT for developers” and ”chatGPT in software development” is generalized as ”chatGPT's impact on software development”. For ”Case study” attribute, different areas of the study has been mentioned in sub-attribute ”chatGPT and software development” and ”chatGPT's impact on software development”. Purpose of this study is discussed in ”Study target” attribute which has a ”impacts of chatGPT in software development” sub-attribute. Study type of the research has been given in Brief Description section of the ”Study type” attribute. Different sources for this research were mentioned in the sub attribute of ”Data source” attribute. The highlights of this study's were mentioned as ”usage”, ”advantage and limitation of chatGPT” sub-attributes of ”Results” attribute. Evaluation of the study is discussed in ”Evaluation” attribute.

Final attributes were further examined to see if they fell in right area or valid.

D. Article Assessment

This part is described in figure 1 (d).And the first two parts of the whole process are described as follows

Attribute assignment In this part, depending on the table II attribute frameworks, each attribute were assigned to relevant articles.Attributes were assigned in such a way they reflected that particular article by answering research questions. During this process, attributes were against the provided description for attribute framework table.Anything that was unmentioned in an article were blank for that attribute field (for example study type).

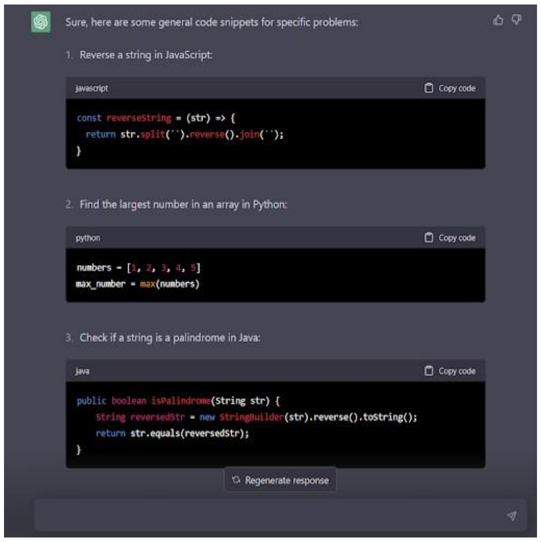
Characterization of the reviewed articles Previous section of assignment of attributes were performed by first author.An article may possess different set of attributes according to another reviewer.To ensure the proper characterization quality of the attribute assignment process, other two author reviewed the assignment.Anything recommended or advised by them were taken into consideration.
The results of review is discussed by attempting and answering our research questions.

IV. OPPORTUNITIES AND CHALLENGES OF CHATGPT

Through answering the research questions opportunities and challenges of chatGPT in software development has been discussed below:

RQ1.How ChatGPT helps in generating code snippets for specific problems?

Using conversational prompt, chatGPT can generate code snippets. This feature is very helpful for developers who want a specefic solution of a code as they can get the specific code snippet from chatGPT by simply describing the problem[15]. To sort an array of numbers in ascending orders, programmer can describe the problem to chatGPT and in response, chatGPT will produce a code snippet in any language of their choice[7]. ChatGPT can advise improvements and flag potential problems due to it's integration for developers. This will favour developers improving quality and minimizing errors in code[1] [17](fig3).



Sure, here are some general code snippets for specific problems:

1. Reverse a string in JavaScript:

```javascript
const reverseString = (str) => {
  return str.split('').reverse().join('');
}
```

2. Find the largest number in an array in Python:

```python
numbers = [1, 2, 3, 4, 5]
max_number = max(numbers)
```

3. Check if a string is a palindrome in Java:

```java
public boolean isPalindrome(String str) {
    String reversedStr = new StringBuilder(str).reverse().toString();
    return str.equals(reversedStr);
}
```

Fig. 3. Code snippets generated by ChatGPT

This usage of chatGPT has an edge as it saves time for developers which used to take a lot of time to do it manually and developers can concentrate more on innovative and complex projects.

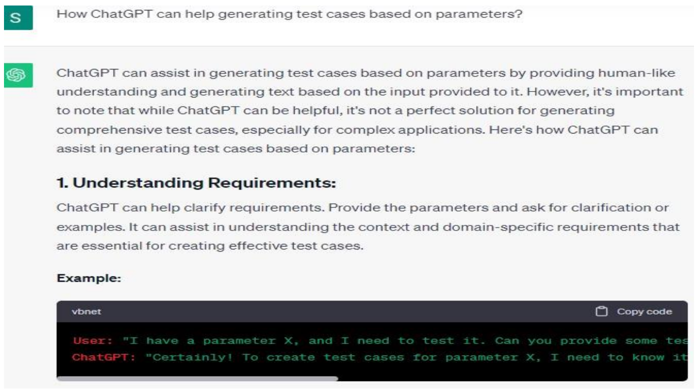
RQ2.How ChatGPT can assist with automating unit tests?

ChatGPT provides great usage for developers in QA testing. Specially for UAT tester it is very useful as it can assist in automating unit testing to make sure expected results are derived from code by catching bugs initially[7][17][18]. If a code with error has been detected in unit testing[15]. To check if chatGPT can identify the error, the problem is asked to chatGPT. In response chatGPT detects the error[1].

For testing automation with the help of chatGPT is beneficial as it lessens human error risk, tests can be run rapidly and repeatedly.

RQ3.How ChatGPT can help generating test cases based on parameters?

One of chatGPT's biggest impact is that it can revolutionalize in generating test cases based on parameters. Usually this process is manually processed considering variables such as boundary condition, data types and edge cases[15]. But chatGPT can effectively produce a vast range of test inputs, taking into account all these variables to confirm that all kinds of scenarios are covered. The outcome is thorough, complete, minimizing the risk of bugs[18]. in fig



How ChatGPT can help generating test cases based on parameters?

ChatGPT can assist in generating test cases based on parameters by providing human-like understanding and generating text based on the input provided to it. However, it's important to note that while ChatGPT can be helpful, it's not a perfect solution for generating comprehensive test cases, especially for complex applications. Here's how ChatGPT can assist in generating test cases based on parameters:

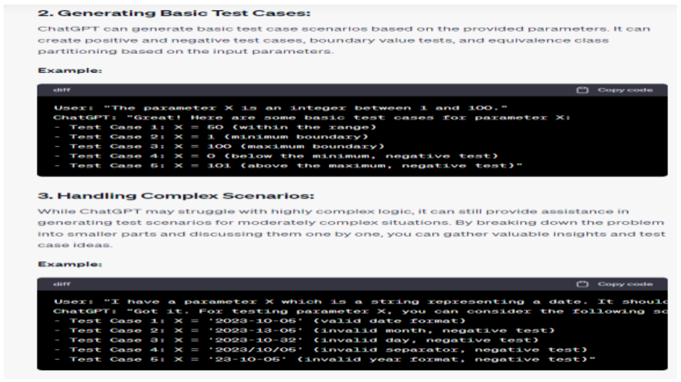
1. Understanding Requirements:

ChatGPT can help clarify requirements. Provide the parameters and ask for clarification or examples. It can assist in understanding the context and domain-specific requirements that are essential for creating effective test cases.

Example:

```vbnet
User: "I have a parameter X, and I need to test it. Can you provide some tes
ChatGPT: "Certainly! To create test cases for parameter X, I need to know it
```

Fig. 4. chatGPT generating test cases based on parameters



2. Generating Basic Test Cases:

ChatGPT can generate basic test case scenarios based on the provided parameters. It can create positive and negative test cases, boundary value tests, and equivalence class partitioning based on the input parameters.

Example:

```diff
User: "The parameter X is an integer between 1 and 100."
ChatGPT: "Great! Here are some basic test cases for parameter X:
- Test Case 1: X = 50 (within the range)
- Test Case 2: X = 1 (minimum boundary)
- Test Case 3: X = 100 (maximum boundary)
- Test Case 4: X = 0 (below the minimum, negative test)
- Test Case 5: X = 101 (above the maximum, negative test)"
```

3. Handling Complex Scenarios:

While ChatGPT may struggle with highly complex logic, it can still provide assistance in generating test scenarios for moderately complex situations. By breaking down the problem into smaller parts and discussing them one by one, you can gather valuable insights and test case ideas.

Example:

```diff
User: "I have a parameter X which is a string representing a date. It shoulc
ChatGPT: "Got it. For testing parameter X, you can consider the following sc
- Test Case 1: X = '2023-10-05' (valid date format)
- Test Case 2: X = '2023-13-05' (invalid month, negative test)
- Test Case 3: X = '2023-10-32' (invalid day, negative test)
- Test Case 4: X = '2023/10/05' (invalid separator, negative test)
- Test Case 5: X = '23-10-05' (invalid year format, negative test)"
```

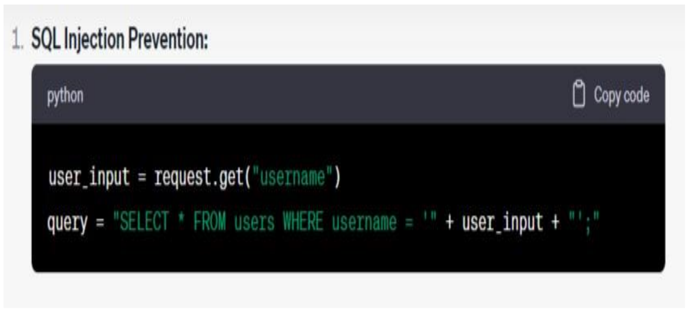
Fig. 5. chatGPT generating test cases based on parameters

4 and 5 chatGPT was asked to produce parameter based test cases and chatGPT gavesome test cases based on parameters.

So chatGPT helps professionals considering all possible inputs to generate test case based on parameters

RQ4.How ChatGPT analyses codes and propose best security methods?

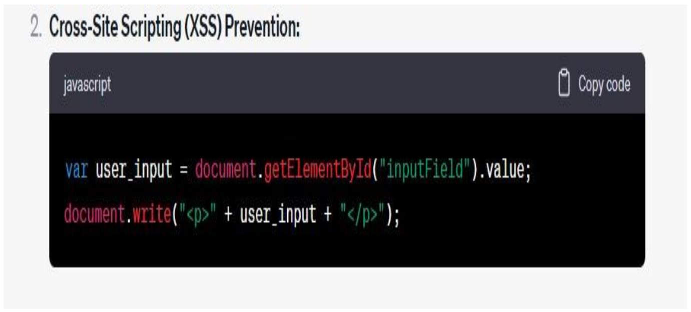
With vast knowledge, it's capability of understanding code pattern and potential vulnerability, chatGPT can analyze codes and deliver suitable recommendations for developing the application's security[3]. It can detect the sensitive area where data is handled, such as personal information or login credentials by only scanning the code. Along with that chatGPT can recommend ways to secure data by encryption methods or safe storage[1]. It is able to detect common weaknesses, like SQL injection or cross-site scripting attacks, fig 6, 7 and suggest guidline on how to mitigate these risks[1]. chatGPT suggested to avoid constructing SQL queries by


1. SQL Injection Prevention:

```python
user_input = request.get("username")
query = "SELECT * FROM users WHERE username = '" + user_input + "';"
```

Fig. 6. security issue with sql injection

concatenating strings. rather suggested to use parameterized queries or prepared statements to prevent SQL injection attacks. To prevent XSS attacks, chatGPT recommennded sanitizing


2. Cross-Site Scripting (XSS) Prevention:

```javascript
var user_input = document.getElementById("inputField").value;
document.write("<p>" + user_input + "</p>");
```

Fig. 7. security issue with cross-site scripting

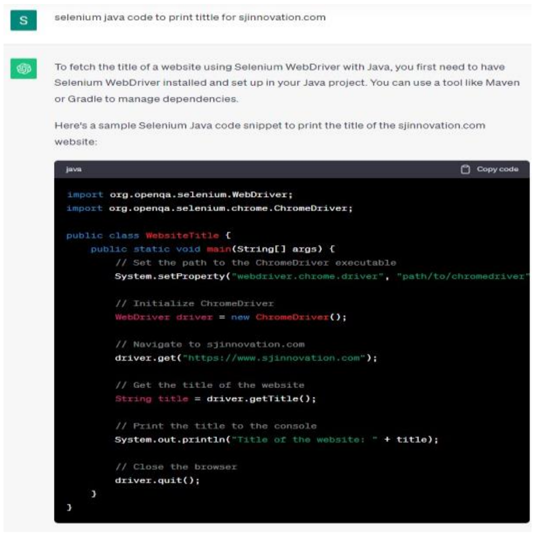
user input before displaying it .

ChatGPT is highly effective in analyzing codes and providing best methods to secure data

RQ5.How can ChatGPT play important role in Automating QA?

ChatGPT helps in many ways for automated testing[3].The advanced language process ability makes it easy for automating repeating tasks including automation of QA. By using chatGPT developers can test the accuracy of their application in real time[16].ChatGPT is particularly effective for developers in QA as it provides immediate results and feedback if their is any issue during development cycle[3]. It can speed up the process by discovering issues and it's solution[3].

In fig 8 chatGPT was asked to provide the tittle of a website using selenium java. Then chatGPT was asked to modify the code using WebDriverManager and testng. ChatGPT provided the code in fig 9, using which the java code was run as testng test class and after successful testing the website tittle was aquired in fig 10. Later the documentation for that code was also asked from chatGPT. chatGPT successfully provided the required documentation in fig 11 and 12



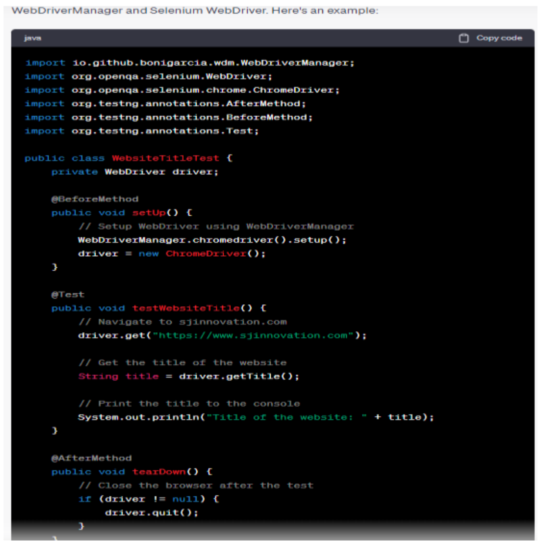
selenium java code to print tittle for sjinnovation.com

To fetch the title of a website using Selenium WebDriver with Java, you first need to have Selenium WebDriver installed and set up in your Java project. You can use a tool like Maven or Gradle to manage dependencies.

Here's a sample Selenium Java code snippet to print the title of the sjinnovation.com website:

```java
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class WebsiteTitle {
    public static void main(String[] args) {
        // Set the path to the ChromeDriver executable
        System.setProperty("webdriver.chrome.driver", "path/to/chromedriver"

        // Initialize ChromeDriver
        WebDriver driver = new ChromeDriver();

        // Navigate to sjinnovation.com
        driver.get("https://www.sjinnovation.com");

        // Get the title of the website
        String title = driver.getTitle();

        // Print the title to the console
        System.out.println("Title of the website: " + title);

        // Close the browser
        driver.quit();
    }
}
```

Fig. 8. qa code


WebDriverManager and Selenium WebDriver. Here's an example:

```java
import io.github.bonigarcia.wdm.WebDriverManager;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

public class WebsiteTitleTest {
    private WebDriver driver;

    @BeforeMethod
    public void setUp() {
        // Setup WebDriver using WebDriverManager
        WebDriverManager.chromedriver().setup();
        driver = new ChromeDriver();
    }

    @Test
    public void testWebsiteTitle() {
        // Navigate to sjinnovation.com
        driver.get("https://www.sjinnovation.com");

        // Get the title of the website
        String title = driver.getTitle();

        // Print the title to the console
        System.out.println("Title of the website: " + title);
    }

    @AfterMethod
    public void tearDown() {
        // Close the browser after the test
        if (driver != null) {
            driver.quit();
        }
```

Fig. 9. modified code

ChatGPT has a huge role to play in QA in upcoming days

RQ6.How effective ChatGPT is in detecting frequent errors? One of ChatGPT's main drawbacks for programmers is that it occasionally provides inaccurate data or responses that are prejudiced[4], both of which have a detrimental effect on the whole quality of the code produced. It's possible that the code is flawed, unusable, or unnecessarily complicated. But in some cases, the solutions it offer lead to more confusion than clarity for some visitors looking for the right answers through chatGPT[11].

Even chatGPT says it has some limitations in detecting frequent coding errors fig 13 .

ChatGPT is not effective enough to detect frequent errors in coding

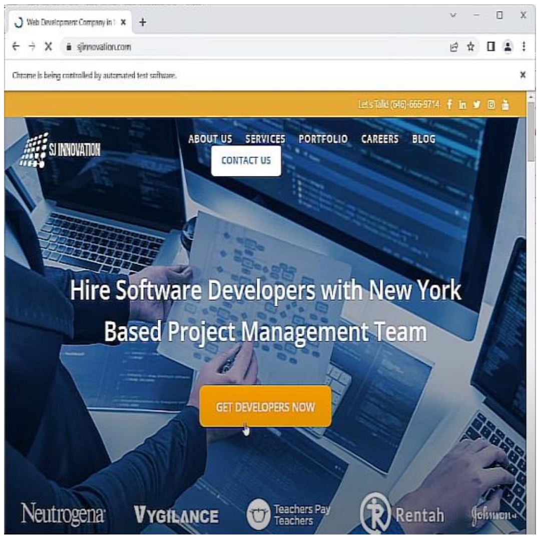
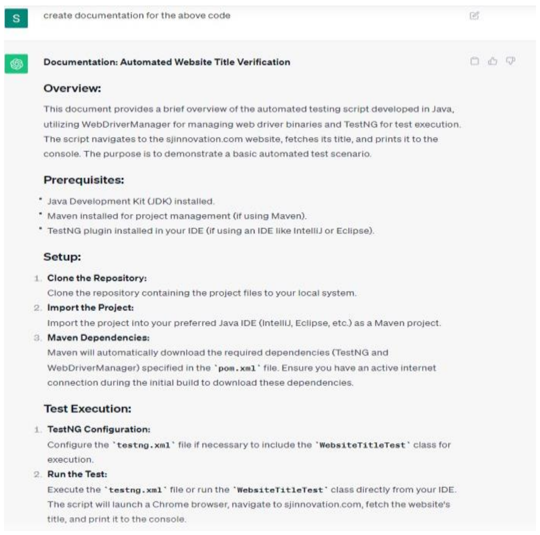

Web Development Company in
sjinnovation.com
Chrome is being controlled by automated test software.
Let's Talk! (646)-665-9714
SJ INNOVATION
ABOUT US
SERVICES
PORTFOLIO
CAREERS
BLOG
CONTACT US
Hire Software Developers with New York Based Project Management Team
GET DEVELOPERS NOW
Neutrogena
VYGILANCE
Teachers Pay Teachers
Rentah

Fig. 10. tittle of website



create documentation for the above code

Documentation: Automated Website Title Verification

Overview:

This document provides a brief overview of the automated testing script developed in Java, utilizing WebDriverManager for managing web driver binaries and TestNG for test execution. The script navigates to the sjinnovation.com website, fetches its title, and prints it to the console. The purpose is to demonstrate a basic automated test scenario.

Prerequisites:

- Java Development Kit (JDK) installed.
- Maven installed for project management (if using Maven).
- TestNG plugin installed in your IDE (if using an IDE like IntelliJ or Eclipse).

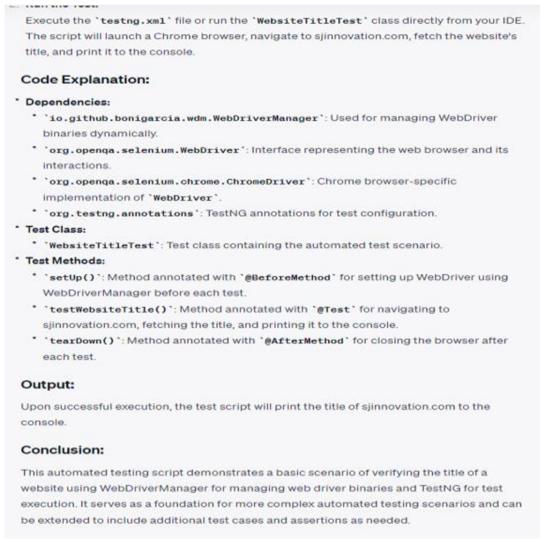
Setup:

1. **Clone the Repository:**
   Clone the repository containing the project files to your local system.
2. **Import the Project:**
   Import the project into your preferred Java IDE (IntelliJ, Eclipse, etc.) as a Maven project.
3. **Maven Dependencies:**
   Maven will automatically download the required dependencies (TestNG and WebDriverManager) specified in the `pom.xml` file. Ensure you have an active internet connection during the initial build to download these dependencies.

Test Execution:

1. **TestNG Configuration:**
   Configure the `testng.xml` file if necessary to include the `WebsiteTitleTest` class for execution.
2. **Run the Test:**
   Execute the `testng.xml` file or run the `WebsiteTitleTest` class directly from your IDE. The script will launch a Chrome browser, navigate to sjinnovation.com, fetch the website's title, and print it to the console.

Fig. 11. documentation



Execute the `testng.xml` file or run the `WebsiteTitleTest` class directly from your IDE. The script will launch a Chrome browser, navigate to sjinnovation.com, fetch the website's title, and print it to the console.

Code Explanation:

- **Dependencies:**
  - `io.github.bonigarcia.wdm.WebDriverManager`: Used for managing WebDriver binaries dynamically.
  - `org.openqa.selenium.WebDriver`: Interface representing the web browser and its interactions.
  - `org.openqa.selenium.chrome.ChromeDriver`: Chrome browser-specific implementation of `WebDriver`.
  - `org.testng.annotations`: TestNG annotations for test configuration.
- **Test Class:**
  - `WebsiteTitleTest`: Test class containing the automated test scenario.
- **Test Methods:**
  - `setUp()`: Method annotated with `@BeforeMethod` for setting up WebDriver using WebDriverManager before each test.
  - `testWebsiteTitle()`: Method annotated with `@Test` for navigating to sjinnovation.com, fetching the title, and printing it to the console.
  - `tearDown()`: Method annotated with `@AfterMethod` for closing the browser after each test.

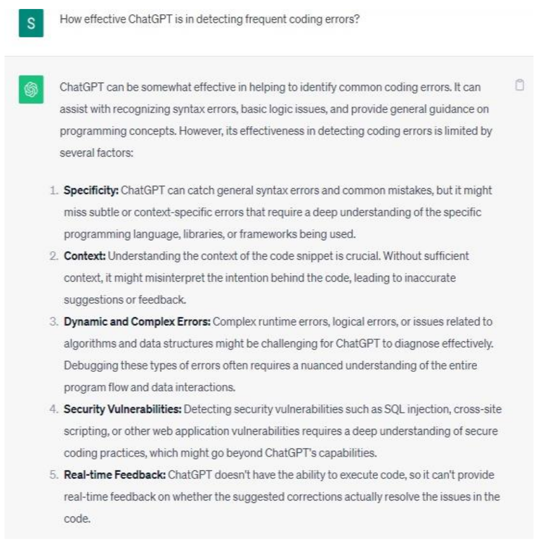
Output:

Upon successful execution, the test script will print the title of sjinnovation.com to the console.

Conclusion:

This automated testing script demonstrates a basic scenario of verifying the title of a website using WebDriverManager for managing web driver binaries and TestNG for test execution. It serves as a foundation for more complex automated testing scenarios and can be extended to include additional test cases and assertions as needed.

Fig. 12. documentation



How effective ChatGPT is in detecting frequent coding errors?

ChatGPT can be somewhat effective in helping to identify common coding errors. It can assist with recognizing syntax errors, basic logic issues, and provide general guidance on programming concepts. However, its effectiveness in detecting coding errors is limited by several factors:

1. **Specificity:** ChatGPT can catch general syntax errors and common mistakes, but it might miss subtle or context-specific errors that require a deep understanding of the specific programming language, libraries, or frameworks being used.
2. **Context:** Understanding the context of the code snippet is crucial. Without sufficient context, it might misinterpret the intention behind the code, leading to inaccurate suggestions or feedback.
3. **Dynamic and Complex Errors:** Complex runtime errors, logical errors, or issues related to algorithms and data structures might be challenging for ChatGPT to diagnose effectively. Debugging these types of errors often requires a nuanced understanding of the entire program flow and data interactions.
4. **Security Vulnerabilities:** Detecting security vulnerabilities such as SQL injection, cross-site scripting, or other web application vulnerabilities requires a deep understanding of secure coding practices, which might go beyond ChatGPT's capabilities.
5. **Real-time Feedback:** ChatGPT doesn't have the ability to execute code, so it can't provide real-time feedback on whether the suggested corrections actually resolve the issues in the code.

Fig. 13. chatGPT's response on detecting frequent errors

RQ7How ChatGPT can provide critical thinking and complexity?

ChatGPT for programmers is unable to support the human abilities of analytical thinking and solving problems. This might cause issues in a number of applications where ChatGPT might respond incorrectly because it is unable to grasp the subtleties of a query[16][7]. The inability of ChatGPT to write code that requires a lot of contexts is another one of the constraints it has for developers[15]. To reach the intended outcome, programmers would have to provide the code with every piece of information possible, which is impractical.

chatGPT is not yet able to provide critical thinking and complexity

RQ8. How capable ChatGPT is to tackle unique problems? ChatGPT for programmers can produce good, straightforward code in a variety of programming languages when given directions, but it is unable to address unusual issues[11]. coders must think about concepts in order to transform them into solutions that satisfy customers' demands in addition to producing programs. It can simulate critical thinking, but the quality of responses still depends on the training data and the specificity of the prompt[12]. ChatGPT for programmers is unlikely to be able to replicate the revolutionary idea exactly as intended, however an experienced programmer can[1].

ChatGPT is unable to tackle unique problems

RQ9.How will ChatGPT perform without out of date data? The fact that ChatGPT has been taught using data from before 2021 is another restriction for programmers. This implies that it is unable to access data or events that have taken place since then[12]. Due to this, ChatGPT's data is 2 years old and insufficient for many programmers, who need up-to-date information to preserve the legitimacy of their work[10][1]. **ChatGPT can not work without data it does not contain**

V. Adopting Factors

Adopt indicates the act of choosing or taking on something, such as a new strategy, method or idea. According to the diffusion of innovation theory, there are five distinct innovation characteristics that decides whether an innovation is successful or not [13]. These characteristics include relative advantage, compatibility, complexity,trialability and observability. diffusion of innovation is the process by which an innovation which might be a new idea, product, practice, philosophy, and so on are adopted slowly by members of a social system. Initially only a few are open to it and adopt its use. But as the time passes the innovation or the product becomes diffused amongst the population until a saturation point is attained.

These five characteristics or adopting factors are being evaluated for the validation of chatGPT as an innovation. Adopting factors implies the factors that influence the adoption of an innovation. In this section, five factors that influence the adoption of chatGPT have been discussed, and each of these factors is at play to a different extent in the adoption of cahtGPT in software development. These factors are,

- Relative Advantage.
- Compatibility.
- Complexity.
- Triability.
- Observability.

These points are discussed bellow

Relative Advantage :

ChatGPT has several relative advantages over traditional methods of customer support, software development, and various other applications, which can be evaluated based on the concept of relative advantage. Relative advantage implies to the degree to which an innovation, like ChatGPT, is perceived as better than the methods or products it replaces[13].

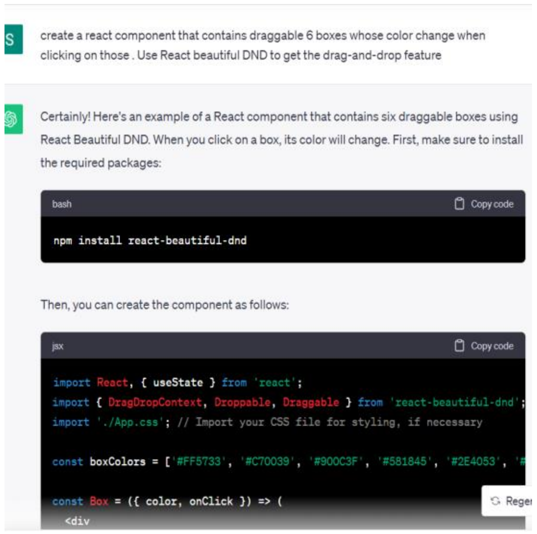
Where in traditional customer support method, it might take significant amount of time, chatGPT can provide instant responses to user queries removing the requirement for customers to wait in queues or wait for replies.

ChatGPT is 24/7 available, therefore everyone has the access to any kind of information. Wherein a human operated service is available only during specific working hours[15].

ChatGPT ensures a huge number of simultaneous conversations making it highly scalable compared to any other services[12]. ChatGPT maintains consistent responses to similar queries, delivering uniformity in the information provided compared to human provided responses which might vary based on their knowledge or mood.

Any human based traditional systems are error prone due to miscommunication or insufficient knowledge but chatGPT generates responses based on patterns in the trained data [15]that is why less error prone than any human.

ChatGPT collects and analyzes data from interactions and offers valuable insights into customer preferences or their frequently asked questions. For example, chatGPT provides code snippets depending upon the problem described to it[11].



create a react component that contains draggable 6 boxes whose color change when clicking on those . Use React beautiful DND to get the drag-and-drop feature

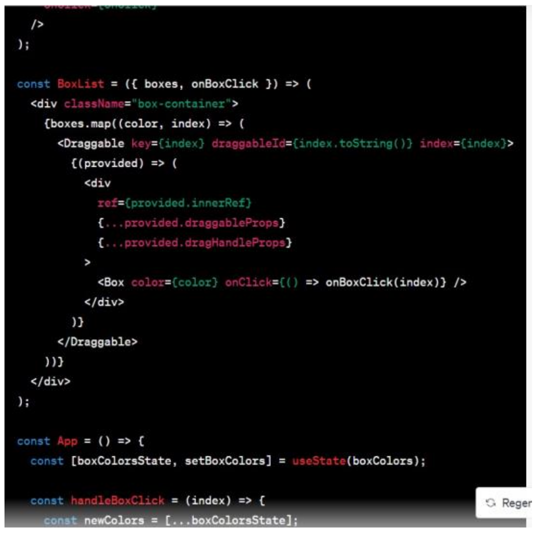
Certainly! Here's an example of a React component that contains six draggable boxes using React Beautiful DND. When you click on a box, its color will change. First, make sure to install the required packages:

```bash
npm install react-beautiful-dnd
```

Then, you can create the component as follows:

```jsx
import React, { useState } from 'react';
import { DragDropContext, Droppable, Draggable } from 'react-beautiful-dnd';
import './App.css'; // Import your CSS file for styling, if necessary

const boxColors = ['#FF5733', '#C70039', '#900C3F', '#581845', '#2E4053', '#

const Box = ({ color, onClick }) => (
  <div
```

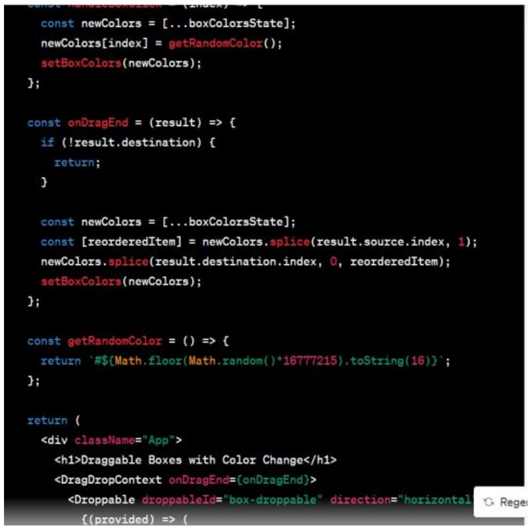
Fig. 14. code generation



```jsx
  />
);

const BoxList = ({ boxes, onBoxClick }) => (
  <div className="box-container">
    {boxes.map((color, index) => (
      <Draggable key={index} draggableId={index.toString()} index={index}>
        {(provided) => (
          <div
            ref={provided.innerRef}
            {...provided.draggableProps}
            {...provided.dragHandleProps}
          >
            <Box color={color} onClick={() => onBoxClick(index)} />
          </div>
        )}
      </Draggable>
    ))}
  </div>
);

const App = () => {
  const [boxColorsState, setBoxColors] = useState(boxColors);

  const handleBoxClick = (index) => {
    const newColors = [...boxColorsState];
```

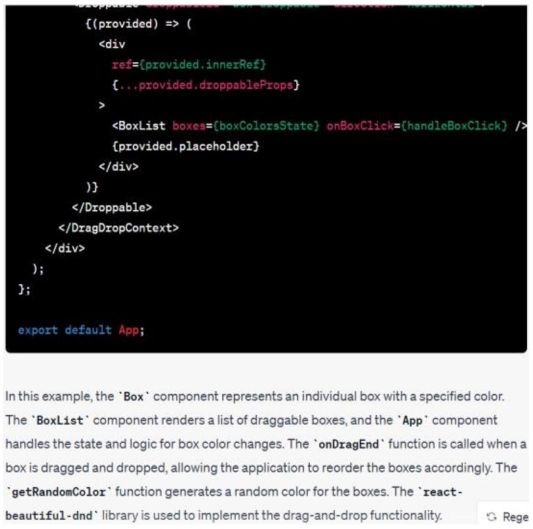
Fig. 15. code generation

Here in fig 14 15 16 17 a specific problem was asked to chatGPT and in answer chatGPT generated the given code. ChatGPT's data-driven approach helps developer to make correct decisions and improve their services based on customer feedback,which is often more challenging to achieve with traditional methods[5]. ChatGPT is also handy in upgrading, finding out probable problems in code. This aids developer finding inaccuracy, refining quality of the respective project[7]. It also provides extra time to focus on more complex and creative projects[8].



```
  const newColors = [...boxColorsState];
  newColors[index] = getRandomColor();
  setBoxColors(newColors);
};

const onDragEnd = (result) => {
  if (!result.destination) {
    return;
  }

  const newColors = [...boxColorsState];
  const [reorderedItem] = newColors.splice(result.source.index, 1);
  newColors.splice(result.destination.index, 0, reorderedItem);
  setBoxColors(newColors);
};

const getRandomColor = () => {
  return `#${Math.floor(Math.random()*16777215).toString(16)}`;
};

return (
  <div className="App">
    <h1>Draggable Boxes with Color Change</h1>
    <DragDropContext onDragEnd={onDragEnd}>
      <Droppable droppableId="box-droppable" direction="horizontal"
        {(provided) => (
```

Fig. 16. code generation



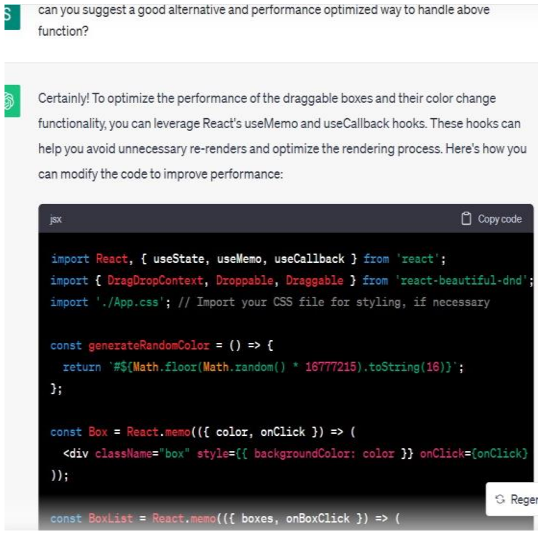
```
        {(provided) => (
          <div
            ref={provided.innerRef}
            {...provided.droppableProps}
          >
            <BoxList boxes={boxColorsState} onBoxClick={handleBoxClick} />
            {provided.placeholder}
          </div>
        )}
      </Droppable>
    </DragDropContext>
  </div>
 );
};

export default App;
```

In this example, the `Box` component represents an individual box with a specified color. The `BoxList` component renders a list of draggable boxes, and the `App` component handles the state and logic for box color changes. The `onDragEnd` function is called when a box is dragged and dropped, allowing the application to reorder the boxes accordingly. The `getRandomColor` function generates a random color for the boxes. The `react-beautiful-dnd` library is used to implement the drag-and-drop functionality.

Fig. 17. code generation

Compatibility: Due to it's consistency of the experiences and needs of potential adopter, chatGPT has a high level of compatibility making it a desirable innovation for various applications[13]. Here's how ChatGPT aligns with the concept of compatibility:

ChatGPT prioritizes user experience as it can understand natural language and provides relevant responses. It's seamless interaction with users makes it flexible[9].
ChatGPT is versatile and adaptable in terms of allowing customized interaction for business[12]. Simultaneously ensuring the innovation aligns with requirements of different user groups.
It is capable of understanding multiple languages, making it compatible with global user's linguistic diversity.
ChatGPT easily integrates with various technology and existing platforms. It is compatible with a huge number off applications,devices and websites[7].Thus chatGPT is incorporated in any business without disruptions.
OpenAI ensures compatibility with users' concerns about privacy and confidentiality by employing encryption and best practices to protect user data. So chatGPT is designed to focus on data privacy and security[12].
OpenAI keeps evolving and boosts ChatGPT based on user feedback and needs. This iterative procedure makes sure that ChatGPT remains compatible with the changing demands and expectations of its users. For example, Given a piece of code as input, chatGPT can recommend other ways to improve performance of the code like lowering operation number, memory usage[15] [14]. These are very useful for developers to write codes more effectively which results in good performance for the project[8].

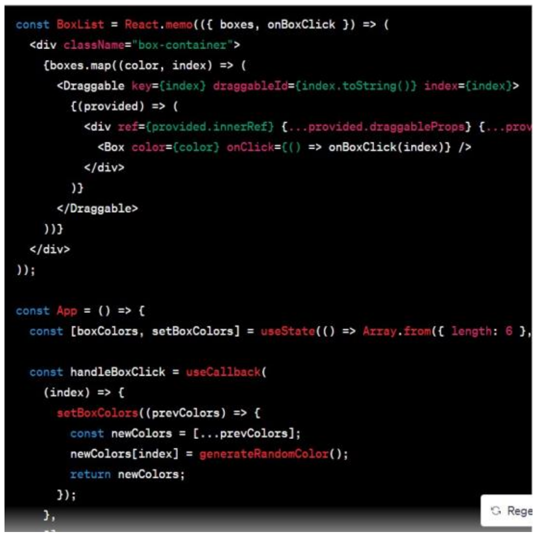


can you suggest a good alternative and performance optimized way to handle above function?

Certainly! To optimize the performance of the draggable boxes and their color change functionality, you can leverage React's useMemo and useCallback hooks. These hooks can help you avoid unnecessary re-renders and optimize the rendering process. Here's how you can modify the code to improve performance:

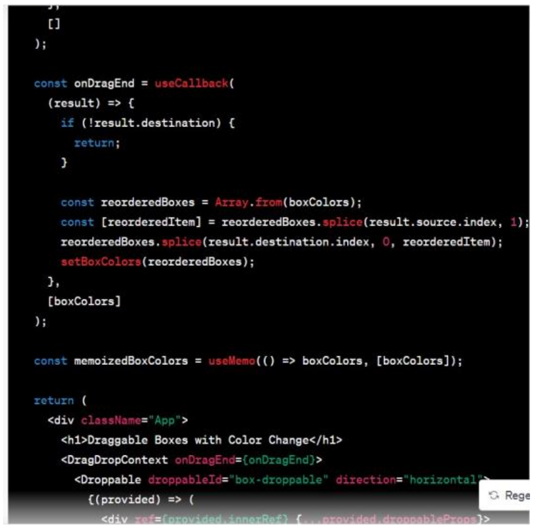
```
import React, { useState, useMemo, useCallback } from 'react';
import { DragDropContext, Droppable, Draggable } from 'react-beautiful-dnd';
import './App.css'; // Import your CSS file for styling, if necessary

const generateRandomColor = () => {
  return `#${Math.floor(Math.random() * 16777215).toString(16)}`;
};

const Box = React.memo(({ color, onClick }) => (
  <div className="box" style={{ backgroundColor: color }} onClick={onClick}
));

const BoxList = React.memo(({ boxes, onBoxClick }) => (
```

Fig. 18. alternate code generation



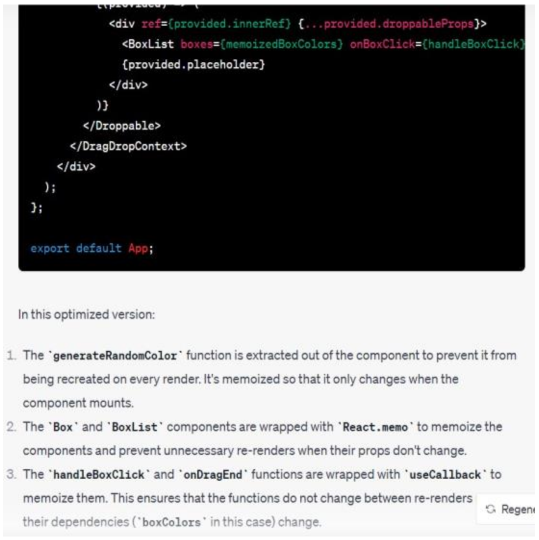
```
const BoxList = React.memo(({ boxes, onBoxClick }) => (
  <div className="box-container">
    {boxes.map((color, index) => (
      <Draggable key={index} draggableId={index.toString()} index={index}>
        {(provided) => (
          <div ref={provided.innerRef} {...provided.draggableProps} {...prov
            <Box color={color} onClick={() => onBoxClick(index)} />
          </div>
        )}
      </Draggable>
    ))}
  </div>
));

const App = () => {
  const [boxColors, setBoxColors] = useState(() => Array.from({ length: 6 },

  const handleBoxClick = useCallback(
    (index) => {
      setBoxColors((prevColors) => {
        const newColors = [...prevColors];
        newColors[index] = generateRandomColor();
        return newColors;
      });
    },
```

Fig. 19. alternate code generation



```
    []
  );

  const onDragEnd = useCallback(
    (result) => {
      if (!result.destination) {
        return;
      }

      const reorderedBoxes = Array.from(boxColors);
      const [reorderedItem] = reorderedBoxes.splice(result.source.index, 1);
      reorderedBoxes.splice(result.destination.index, 0, reorderedItem);
      setBoxColors(reorderedBoxes);
    },
    [boxColors]
  );

  const memoizedBoxColors = useMemo(() => boxColors, [boxColors]);

  return (
    <div className="App">
      <h1>Draggable Boxes with Color Change</h1>
      <DragDropContext onDragEnd={onDragEnd}>
        <Droppable droppableId="box-droppable" direction="horizontal"
          {(provided) => (
            <div ref={provided.innerRef} {...provided.droppableProps}>
```

Fig. 20. alternate code generation



```
            <div ref={provided.innerRef} {...provided.droppableProps}>
              <BoxList boxes={memoizedBoxColors} onBoxClick={handleBoxClick}
              {provided.placeholder}
            </div>
          )}
        </Droppable>
      </DragDropContext>
    </div>
  );
};

export default App;
```

In this optimized version:

1. The `generateRandomColor` function is extracted out of the component to prevent it from being recreated on every render. It's memoized so that it only changes when the component mounts.
2. The `Box` and `BoxList` components are wrapped with `React.memo` to memoize the components and prevent unnecessary re-renders when their props don't change.
3. The `handleBoxClick` and `onDragEnd` functions are wrapped with `useCallback` to memoize them. This ensures that the functions do not change between re-renders their dependencies (`boxColors` in this case) change.

Fig. 21. alternate code generation

Alternate code for **Fig 14** was asked to chatGPT in fig 18 and chatGPT provided alternate code that would bring better performance in fig 18 19 20 21.

ChatGPT is also compatible with educational needs of students, educators[7]. It enhances its value by providing tutoring, explanations, and learning resources.

ChatGPT is significantly compatible with customer sup- port needs. Customer inquiry, product information and troubleshooting assistance offer etc are handled by chatGPT for business purposes.

In summary, ChatGPT's compatibility is evident through its user-centric design,flexibility, adaptability, linguistic capabilities,integration capability,security measures and various aspects of education, business, and customer support

Complexity: ChatGPT is designed to be user-friendly, flexible minimizing complexity for its users. Under it's sophisticated technology, it is abstracted away to deliver a seamless experience. Here's how ChatGPT's complexity can be understood:

ChatGPT has a very user friendly interface that is quite easy to interact with. User can put texts and get responses. User can have a natural conversation on chatGPT's interface the way most people are familiar with[9]. Users can communicate in the same way they would with a human, simplifying the interaction process.

Users do not need to be a technical expertise to use ChatGPT. ChatGPT lifts away the technical complexities, allowing non-technical users get the most advantage from chatGPT. It is not like some other AI systems that require a deep understanding of technical or machine learning concepts.

As ChatGPT comes pre-trained on huge datasets[8],it produces human-like responses out of the box. So users do not require to understand complexity of training models, data preprocessing, or fine-tuning, which simplifies user experience.

ChatGPT can easily understand context. Users do not need to rephrase their queries or add extensive context to make chatGPT understand. ChatGPT maintains the conversation flow natural and feasible[9].

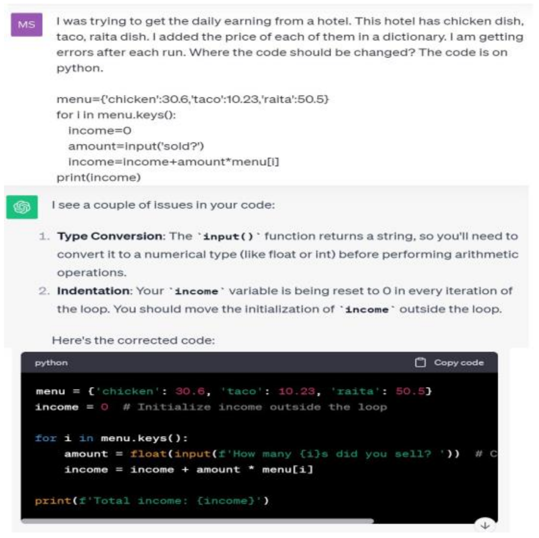
The integration of ChatGPT into various applications is simplified by APIs and documentation provided by OpenAI. OpenAI's community forums provide guidelines or suggestions to users and developers regarding troubleshooting,fixing bugs or exploring advanced features[16]. ChatGPT provides available resources or solutions to help them navigate any challenges they might encounter, lessening the difficulty of using ChatGPT. For example, chatGPT will inspect a piece of code, suggest developers some actions to find and correct bugs[17][7]. By this, developers can spend fewer times in identifying bug's root cause in fig 22[18][14].

In summary, ChatGPT's complexity is reduced through its instinctive interface, requirement for non-technical knowledge, pre-trained models, accessible APIs and many other aspects. These factors jointly ensure that ChatGPT is easy to use for a broad range of users, regardless of their technical knowledge level.

Triability: OpenAI has made it accessible for users to test and evaluate the technology before making a commitment to adopt it fully. Here's how ChatGPT's triability is demonstrated:

OpenAI provides an online platform the OpenAI Playground where users can check ChatGPT's accuracy in real-time without any cost. This allows users to experiment with the technology, understand its potentials and its usefulness before making any financial commitment[3].

Free access and trials to ChatGPT is offered by OpenAI. Through its API, users get to explore it's functionalities and



MS I was trying to get the daily earning from a hotel. This hotel has chicken dish, taco, raita dish. I added the price of each of them in a dictionary. I am getting errors after each run. Where the code should be changed? The code is on python.

```
menu={'chicken':30.6,'taco':10.23,'raita':50.5}
for i in menu.keys():
   income=0
   amount=input('sold?')
   income=income+amount*menu[i]
print(income)
```

I see a couple of issues in your code:

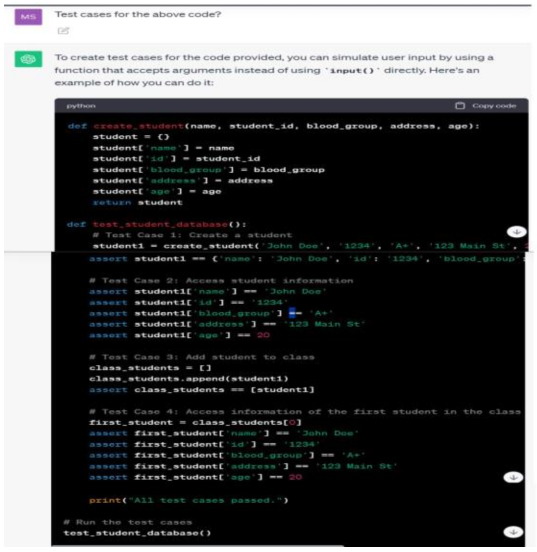
1. **Type Conversion**: The `input()` function returns a string, so you'll need to convert it to a numerical type (like float or int) before performing arithmetic operations.
2. **Indentation**: Your `income` variable is being reset to 0 in every iteration of the loop. You should move the initialization of `income` outside the loop.

Here's the corrected code:

```python
menu = {'chicken': 30.6, 'taco': 10.23, 'raita': 50.5}
income = 0  # Initialize income outside the loop

for i in menu.keys():
    amount = float(input(f'How many {i}s did you sell? '))  # C
    income = income + amount * menu[i]

print(f'Total income: {income}')
```

Fig. 22. avoiding bugs

applications[15].Before any financial investment, These trials helps individuals and businesses to test the technology in their specific use cases.
OpenAI provides free access of chatGPT to students, researchers, and educators for experimentation and academic purposes. So that they can continue to learn and research without any cost[12].
OpenAI offers comprehensive documentation for developers with tutorials and examples[5]. Resources provided by OpenAI allows developers to experiment and incorporate chatGPT into their applications[1], and test its functionalities in a constrained environment before applying on a full scale adoption. For example, if a developer gives a code snippet of a classroom (20 students) with their name,blood group, id, address and asks chatGPT to create a database using python, it can create test inputs with that [15]in fig 23.



MS Test cases for the above code?

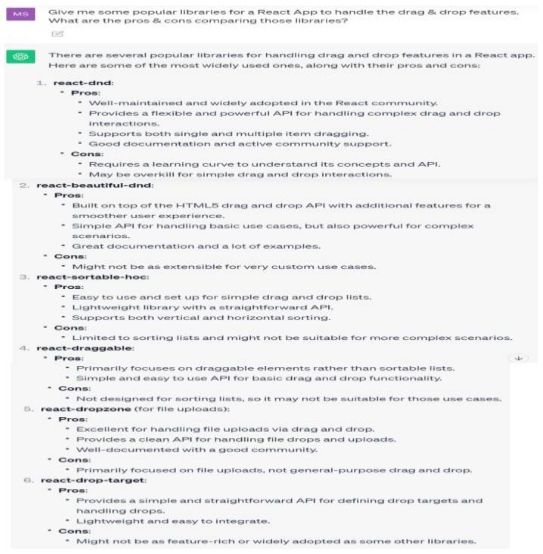
To create test cases for the code provided, you can simulate user input by using a function that accepts arguments instead of using `input()` directly. Here's an example of how you can do it:

```python
def create_student(name, student_id, blood_group, address, age):
    student = {}
    student['name'] = name
    student['id'] = student_id
    student['blood_group'] = blood_group
    student['address'] = address
    student['age'] = age
    return student

def test_student_database():
    # Test Case 1: Create a student
    student1 = create_student('John Doe', '1234', 'A+', '123 Main St',
    assert student1 == {'name': 'John Doe', 'id': '1234', 'blood_group'

    # Test Case 2: Access student information
    assert student1['name'] == 'John Doe'
    assert student1['id'] == '1234'
    assert student1['blood_group'] == 'A+'
    assert student1['address'] == '123 Main St'
    assert student1['age'] == 20

    # Test Case 3: Add student to class
    class_students = []
    class_students.append(student1)
    assert class_students == [student1]

    # Test Case 4: Access information of the first student in the class
    first_student = class_students[0]
    assert first_student['name'] == 'John Doe'
    assert first_student['id'] == '1234'
    assert first_student['blood_group'] == 'A+'
    assert first_student['address'] == '123 Main St'
    assert first_student['age'] == 20

    print("All test cases passed.")

# Run the test cases
test_student_database()
```

Fig. 23. test case

ChatGPT promotes the engagement of a community of developers and researchers by providing a platform for them to share their experiments and gather insights into how ChatGPT can be trialed and tested in various contexts[4].
OpenAI asks it's users to provide feedback on problematic model outputs which allows users to point out issues they encounter during their trials[17]. This improves ChatGPT's performance and reliability.
Hackathons and competitions are occasionally hosted by OpenAI that allow developers to showcase their innovative uses of ChatGPT to experiment with the technology and present their creations, fostering a culture of experimentation and innovation.
So chatGPT's features demonstrate triability. These features empower users to test, experiment, evaluate, and explore ChatGPT in several scenarios before adopting it fully, fostering a flexible process.

Observability: The extent to which the innovation provides tangible and measurable outcomes in various applications, refers to ChatGPT's observability[13]. Here's how ChatGPT's observability can be understood:
ChatGPT provides tangible effects on user experi- ence on websites and applications[15]. Higher conversion rates,increased user engagement and longer time spent on platforms are tangible outcomes of enhanced user experience due to chatGPT.
Entrepreneurs and developers can build products and ser- vices using resources and necessary information provided by ChatGPT[7]. New business opportunities and tangible products are created with the help of chatGPT.
ChatGPT can assist in integrating applications and services developed using various programming languages, converting into a tangible enhancement to software projects[14]. This model is utilized by developers using programming languages like Python, JavaScript, or others, enabling consistent communication between ChatGPT and other software components. For example, ChatGPT can be asked technical questions related to a programming language, it would then respond with an appropriate response in fig 24.
ChatGPT has impact on online education and learning platforms.For personal and professional development, The knowledge and skills acquired by users through these platforms have tangible impacts on their respective sectors of work[11].
Job opportunities are being produced in AI development, data analysis and customer service. ChatGPT's service and applications are contributing to the economy by creating these jobs[7]. These job opportunities are tangible results of ChatGPT's usage in the market.
Businesses make informed choices about their products and service based on the interactions and feedback received from chatGPT[12]. This leads to improvement and growth in their businesses such as increased sales, improved customer satisfaction, and cost saving. Thus ChatGPT can have tangible effects on businesses. These effects have real-world monetary



MS Give me some popular libraries for a React App to handle the drag & drop features. What are the pros & cons comparing those libraries?

There are several popular libraries for handling drag and drop features in a React app. Here are some of the most widely used ones, along with their pros and cons:

1. **react-dnd**:
   - **Pros**:
     - Well-maintained and widely adopted in the React community.
     - Provides a flexible and powerful API for handling complex drag and drop interactions.
     - Supports both single and multiple item dragging.
     - Good documentation and active community support.
   - **Cons**:
     - Requires a learning curve to understand its concepts and API.
     - May be overkill for simple drag and drop interactions.
2. **react-beautiful-dnd**:
   - **Pros**:
     - Built on top of the HTML5 drag and drop API with additional features for a smoother user experience.
     - Simple API for handling basic use cases, but also powerful for complex scenarios.
     - Great documentation and a lot of examples.
   - **Cons**:
     - Might not be as extensible for very custom use cases.
3. **react-sortable-hoc**:
   - **Pros**:
     - Easy to use and set up for simple drag and drop lists.
     - Lightweight library with a straightforward API.
     - Supports both vertical and horizontal sorting.
   - **Cons**:
     - Limited to sorting lists and might not be suitable for more complex scenarios.
4. **react-draggable**:
   - **Pros**:
     - Primarily focuses on draggable elements rather than sortable lists.
     - Simple and easy to use API for basic drag and drop functionality.
   - **Cons**:
     - Not designed for sorting lists, so it may not be suitable for those use cases.
5. **react-dropzone** (for file uploads):
   - **Pros**:
     - Excellent for handling file uploads via drag and drop.
     - Provides a clean API for handling file drops and uploads.
     - Well-documented with a good community.
   - **Cons**:
     - Primarily focused on file uploads, not general-purpose drag and drop.
6. **react-drop-target**:
   - **Pros**:
     - Provides a simple and straightforward API for defining drop targets and handling drops.
     - Lightweight and easy to integrate.
   - **Cons**:
     - Might not be as feature-rich or widely adopted as some other libraries.

Fig. 24. Programming language related query

implications for organizations, making ChatGPT's influence tangible. The expanding productivity and saved time are also tangible outcomes of chatGPT in various application.
ChatGPT can minimize the workload on employees, by handling mundane and repetitive tasks. This workload reduction translates into tangible benefits like stress reduction, better balanced work-life. etc.
Though chatGPT exists in the digital space, its applications and the results of its usage have tangible effects on developers,individuals, business, education and society at large. These tangible outcomes illustrate the practical value of ChatGPT in several real-world scenarios.

VI. Discussion

Generative AI, such as ChatGPT, undoubtedly has a significant impact on how organizations develop applications, from allowing quicker and more effective development cycles to enhancing user experiences[12].In this study, Systematic Literature Review methodology has been adopted to discuss the impact of chatGPT on software development. The study answered research questions alongside discussing results and findings. For this, a protocol has been followed. The protocol involves a step-by-step procedure. To fulfill the motive, 9 research questions had been generated. The questions fall into these categories - Usage, Benefit, Efficiency, Limita- tions. Usage includes finding out how ChatGPT is used in developing software or in software generally. In benefit, the target was to find out ChatGPT's advantages in software development field. In efficiency, the motive was to figure out ChatGPT's efficiency in software development. Limita- tion includes its drawbacks. Selected articles' provided data were assigned to attribute set to answer research questions. Following that,Those research questions and their results have been discussed. ChatGPT's validation as an innovation or method has been discussed with 5 adopting factors. Chat- GPT's *Relative advantage*,*Compatibility*,*Complexity*,*Triability* and *Observability* as a method in software development was reviewed.This review paper provides an explicit idea about the impact of chatGPT on software development and specific rationale behind the incorporation of chatGPT in software development or software engineering in general.

A. *Main finding*

For the time being, Chat GPT can help bridge gaps and speed the implementation of solutions within the process of developing software, but developers will still be required to drive relevant experiences.
ChatGPT is effective in generating code snippets, detecting errors within the code but unable to detect frequent errors.For testing automation, unit testing or in creating parameter based test case, chatGPT is beneficial as it lessens human error risk and tests can be run rapidly and repeatedly. ChatGPT can even analyze code to propose some security methods but chatGPT can not lacks critical thinking and stumbles if there is any unique problem. ChatGPT is very beneficial in fixing bugs and thus improves code quality. ChatGPT lacks the capacity to comprehend the human context of computers, which is required for good programming[16]. More information about the intent of the software and the people who will use it can be added by software engineers. It's not merely a collection of applications thrown together using recycled code. Engineering necessitates several factors that AI cannot replace, such as context, which makes it very hard for AI to put into a single model, train that model, and add the predictive skills of people who understand what will be required in the next few years[17]. There are many big-picture choices that are unique to each organization that AI is unlikely to be able to manage[10]. One paradigm shift is the transition from keyword-based search engines to those that interpret natural language searches and provide helpful replies[3][2]. ChatGPT may produce boilerplate or suggested example code for problems much quicker than any developer can write and experiment with code from scratch by submitting questions in normal conversational English[7]. It will give valuable tools that will remove repetitive activities and expedite app development time to market, as well as allow developers to accelerate the repeated decisions that designers must make.

B. *Limitation*

From RQ6 of "Opportunities and Challenges" section, it has been found that chatGPT has some limitations in detecting frequent coding errors.It occasionally provides inaccurate data or responses that are prejudiced, both of which have a detrimental effect on the whole quality of the code produced[6].
From RQ7 it has been proved that chatGPT is not yet able to provide critical thinking and complexity. The inability of ChatGPT to write code that requires a lot of contexts is another one of the constraints it has for developers.
RQ8 reveals that ChatGPT is unable to tackle unique problems. Straightforward code in a variety of programming languages is produced by chatGPT when given directions, but it is unable to address unusual issues.

From RQ9 it has been proved that ChatGPT can not work without data it does not contain.

However, utilizing generative AI to implement systematic modifications to workflows is difficult. The authors of ”Power and Prediction: The Disruptive Economics of Artificial Intelligence” equate point solutions (such as discovering code samples) with AI system solutions that will necessitate more significant adjustments. ChatGPT still requires real-time feedback to guarantee that the model is operating correctly[8]. These models' triumphs and failures will be determined by the data and people behind them. It's simple to understand how it might help identify coding examples or improve code quality[18]. However, before incorporating generative AI into their application, product executives and their teams of agile developers ought to verify and test its application cases[10]. The possibility of an uncontrolled AI providing false or incomplete material is at best unpleasant and at worst extremely costly, especially when utilized for customer assistance or brand representation[12][4][6]. Although it may be tempting to let AI generate content on its own, such as an unsupervised chatbot, brands are going to quickly understand that in order to manage this threat, they must employ an integrated approach in which humans and AI collaborate[15]. Chat GPT is more than just a pretty face, but like with any novel innovation, software developers and architects will need to test where, when, and how to leverage generative AI capabilities[16].

C. Future scope

ChatGPT will play crucial role in software development as it is capable assisting human software architects by providing ideas,suggestions and feedback on design decision[5]. ChatGPT's ability to generate code snippets, error detection and automated bug fixing capability will improve productivity as well as save valuable time of developers[4]. ChatGPT has the capability to change the way human think about web design and development[9]. Because of its capacity to recognize and create natural language, it is a great tool for developing conversational interfaces, customizing material, and producing interesting content[12][18]. ChatGPT is poised to play a more vital role as organizations explore methods to keep ahead of the curve and give the greatest user experience to their consumers.

VII. Conclusion

The advancement of AI technology is unquestionably beneficial for the software development sector[4]. Unquestionably, ChatGPT is a ground-breaking AI technology that will be extremely useful to software developers. By having the ability to generate code snippets, respond to queries, and generate documents in a matter of seconds, efficiency can undoubtedly be increased[15][18]. Developers do not need to be special- ists in a specific programming language or framework to participate since ChatGPT can understand natural language commands[2]. As a result, the development community may broaden and become more inclusive, and fresh ideas and per- spectives might be offered. Although ChatGPT is a remarkable accomplishment, the human brain still has the upper hand. For a while to come, humans will still be crucial to the software development process, and innovations in artificial intelligence will make them better programmers.

References

[1] Boosting efficiency for developers coders with the help of chatgpt! Available at https://sjinnovation.com/boosting-efficiency- developers-coders-help-chatgpt: :text=Developers(2023/02/15).

[2] A. Ahmad, M. Waseem, P. Liang, M. Fahmideh, M. S. Aktar, and T. Mikkonen. Towards human-bot collaborative software architecting with chatgpt. In *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering*, pages 279–285, 2023.

[3] Akbar, M. Azeem, Khan, A. Ali, Liang, and Peng. Ethical aspects of chatgpt in software engineering research. *arXiv preprint arXiv:2306.07557*, 2023.

[4] A. S. Bale, Y. R. Vada, B. E. Oshiojum, U. K. Lakkineni, C. Rao, K. Venkatesh, and I. Rani. Chatgpt in software development: Methods and cross-domain applications. *International Journal of Intelligent Systems and Applications in Engineering*, 11(9s):636–643, 2023.

[5] A. Beganovic, M. A. Jaber, and A. A. Almisreb. Methods and applications of chatgpt in software development: A literature review. *Southeast Europe Journal of Soft Computing*, 12(1):08–12, 2023.

[6] S. Bordt and U. Luxburg. Chatgpt participates in a computer science exam, 2023.

[7] M. Fraiwan and N. Khasawneh. A review of chatgpt applications in education, marketing, software engineering, and healthcare: Benefits, drawbacks, and research directions. *arXiv preprint arXiv:2305.00237*, 2023.

[8] D. N. GAMAGE. 7 ways chatgpt can help developers. Available at https://cult.honeypot.io/reads/how-can-chatgpt-help-developers/ (2023/03/07).

[9] A. Haleem, M. Javaid, and R. P. Singh. An era of chatgpt as a significant futuristic support tool: A study on features, abilities, and challenges. *BenchCouncil Transactions on Benchmarks, Standards and Evaluations*, 2(4):100089, 2022.

[10] D. Ham. Is human software development doomed with chatgpt. Available at https://www.getfishtank.com/blog/how-chatgpt-will-affect-software-development: :text=Limited(2023/01/07).

[11] S. Jalil, S. Rafi, T. D. LaToza, K. Moran, and W. Lam. Chatgpt and software testing education: Promises & perils. In *2023 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 4130–4137. IEEE, 2023.

[12] D. Kalla and N. Smith. Study and analysis of chat gpt and its impact on different fields of study. *International Journal of Innovative Science and Research Technology*, 8(7), 2023.

[13] June Kaminski. Diffusion of innovation theory. *Canadian Journal of Nursing Informatics*, 6(2):1–6, 2011.

[14] A. Kashefi and T. Mukerji. Chatgpt for programming numerical methods. 03 2023.

[15] N. Nascimento, P. Alencar, and D. Cowan. Comparing software developers with chatgpt: An empirical investigation. *arXiv preprint arXiv:2305.11837*, 2023.

[16] I. Sacolick. Chatgpt and software development. Available at https://www.infoworld.com/article/3689172/chatgpt-and-software-development.html (2023/02/27).

[17] D. Sobania, M. Briesch, C. Hanna, and J. Petke. An analysis of the automatic bug fixing performance of chatgpt. *arXiv preprint arXiv:2301.08653*, 2023.

[18] N. M. S. Surameery and M. Y. Shakor. Use chatgpt to solve programming bugs. *International Journal of Information technology and Computer Engineering*, 3(01), 2023.