Iraqi Journal of Science, 2021, Vol. 62, No. 4, pp: 1349-1363
DOI: 10.24996/ijs.2021.62.4.31

ISSN: 0067-2904

Intelligent Agent Services in Electronic Libraries

Reem.A.Qader *, Naji.M.Sahib
College of Science-Depertment of Computer Science-University of Diyala, Baghdad, Iraq

Received: 30/5/2020 Accepted: 22/7/2020

Abstract
Global services with an agent or a multi-agent system are a promising and new research area. However, several measures have been proposed to demonstrate the benefits of agent technology by supporting distributed services and applying smart agent technology in web dynamics. This paper is designed to build a Semantic Web on the World Wide Web (WWW) to enhance the productivity of managing electronic library applications, which poses a problem to researchers and students, represnted by the process of exchanging books from e-libraries, where the process is slow or the library needs large system data.
In this work, we found a solution to this problem by using agent technology based on the WebSocket, where any user can use this library to achieve fast communication and high information based on the e-library. The model will be simple and small. In addition, the library does not need an employee responsible for entering information into the library database, as the Firebase includes a cloud-based database that synchronizes data across every client in Real-time and supplies offline functionality. Any researcher can access the form by logging in. This application is installed on the central library server and every user who uses this library reaches a quick result, as conducted and proven in the current work.
On average, the time required to process 10 requests using HTTP was about 25 ms, while it was 19 ms when Socket.io was utilized for the processing of 100, 500, and 1000 requests, time values were 168 ms, 779 ms, and 1520 ms using HHTP and 30 ms, 102 ms, and 172 ms using Socket.io, respectively. Therefore, the proposed model which employes Websockets is about 5 to 7 times faster than HTTP.

Keywords: Intelligent agent, Multi-agent systems, distributed services, WebSocket, e-library.

خدمات الوكيل الذكي في المكتبة الإلكترونية

ريم عادل قادر*، ناجي مطر سحيب
قسم علوم الحاسبات، كلية العلوم، جامعة ديالى، بغداد ، العراق

الخلاصة
تعد الخدمات العالمية للوكيل أو نظام متعدد الوكلاء مجال بحث جديد و واعد. ومع ذلك ، تم اقتراح العديد من التدابير لإثبات فوائد تقنية الوكيل من خلال دعم الخدمات الموزعة وتطبيق تكنولوجيا الوكيل الذكي في ديناميكيات الويب.
هذا البحث عبارة عن تصميم لبناء شبكة دلالية على شبكة الويب العالمية لتعزيز إنتاجية إدارة تطبيقات المكتبة الإلكترونية ، حيث توجد مشكلة يواجهها الباحثون والطلاب ، وهي عملية تبادل الكتب من المكتبات الالكترونية

*Email: rkhwedim2@gmail.com

حيث تكون بطيئة أو أن المكتبة تحتاج إلى بيانات نظام كبيرة. في هذا العمل تم ايجاد حلاً لهذه المشكلة بأستخدام تقنية الوكيل المستندة الى(WebSocket) ، اذ يمكن لأي مستخدم استخدام هذه المكتبة للحصول على اتصال سريع ومعلومات عالية بناءً على المكتبة الالكترونية.
سوف يكون النموذج بسيطاً وصغيرا ، فضلا عن ذلك ، لا تحتاج المكتبة الى موظف مسؤول لادخال المعلومات في قاعدة بيانات المكتبة اذ تم وضعها في (Firebase) حيث إنها قاعدة بيانات قائمة على السحابة تقوم بمزامنة البيانات عبر كل عميل في الوقت الحقيقي ، وتوفر وظائف دون اتصال.
يمكن لأي باحث الوصول إلى النموذج عن طريق تسجيل الدخول. ويتم تثبيت هذا التطبيق على خادم المكتبة المركزية ويحصل كل مستخدم يستخدم هذه المكتبة على نتيجة سريعة وتم إجراؤها وإثباتها في العمل الحالي. في المتوسط استغرق 10طللبات (HTTP) حوالي 25 ملي ثانية وطلبات (Socket.io) 19 ملي ثانية, بينما استغرق 100 طلب (HTTP) حوالي 168 ملي ثانية بينما (Socket.io) 30 ملي ثانية . وايضا, تم حساب 500 طلب (HTTP) استغرق حوالي 779 ملي ثانية وطلب (Socket.io) 102 ملي ثانية , وكما يستغرق 1000 طلب (HTTP) حوالي 1520 ملي ثانية اما طلب (Socket.io) حوالي 172 ملي ثانية. لذلك, فان النموذج المقترح (WebSocket) اسرع من 5 الى 7 مرات من (HTTP) العادي.

1. Introduction

In 1996, Broadcom Inc. Ltd. (Ireland) started a .research collaboration with the Department of Computer Science at Trinity College Dublin to use the new technology of intelligent agents (IA) for the enforcement of telecommunications. Intelligent agents are currently regarded as an important topic in information systems. [1].

An agent is anything that can be viewed as perceiving its environment through sensors and, acting upon that environment through actuators [2]. A general representation of the role of IA is shown in Figure-1.

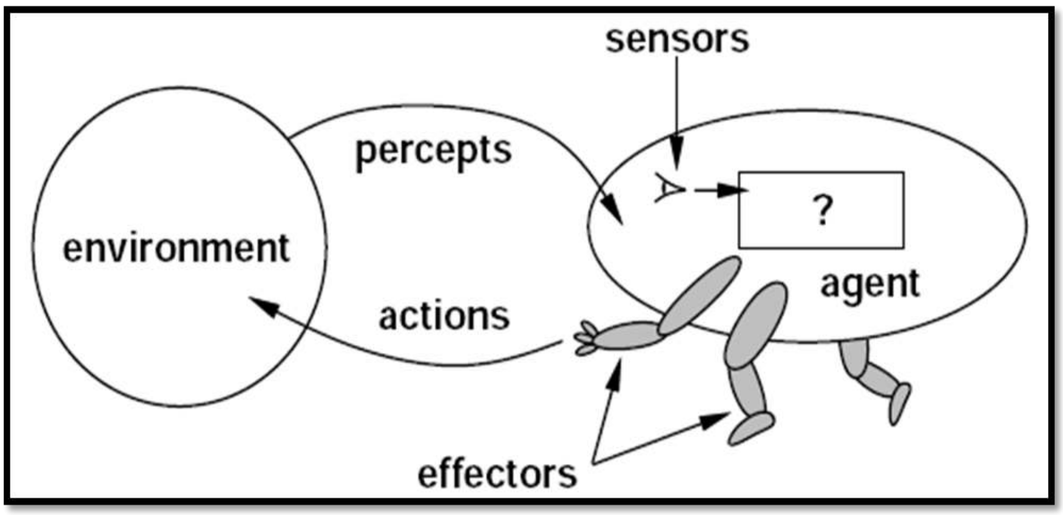


Figure 1-Agent's interaction with environments through sensors and effectors [2].

Applications of the intelligent agents can be classified according to the type of agent, the domain of application, and the technologies used to implement the agent. The main applications of intelligent agents include industrial, commercial, and medical applications, as well as entertainment. [3].

An intelligent agent is an autonomous computer system that has a smart technique that suites the environment to solve complicated problems. Any agent interacts with the environment through an agent life cycle, which means making observations by sensing, deciding, and taking an action. To solve the most complicated problems that cannot be solved through a single smart entity, the modern approach agent services were developed. The distinguishing features of agents include having an individual viewpoint and local control to solve a part of the problem. Meanwhile, all agents decentralize data and asynchronous processing in the agent's society. Smarter entities within a society can change the way of interaction between individuals belonging to this society. Human beings, intuitively, follow smarter people and adapt their behavior. An agent society is a society of software modules with socialization capabilities and two types of behaviors; first, the individual behavior that launches from knowledge base built upon the creation of the entity in addition to reforms that occur due to the interaction with the environment. Second, the social behavior, which is the dominant behavior [4]. Many software systems are generally distributed, frameworks that are, run on a broadly integrated group of processors of the network [5,6]. Numerous research attempts have been accomplished, to test distributed frameworks that have issues such as security, concurrency, timing, and controllability. Others [7, 8, 9, .5] have concentrated their investigations on moving forward where the elucidation of the testing was arranged to make it more adaptive and faster.

1.2The Artificial Intelligence

The subject of artificial intelligence (AI) is rooted in various research disciplines such as philosophy [10, 11] computer science [12, 13], and important new studies [14]. In this paper, the proposed model fundamentally concentrates on the domain of computer science, as it is of the utmost importance in intelligence agents to AI.Artificial Intelligent research .is divided, into various, research .streams. These vary from the topic of AI application (acting vs. thinking) to the decision making type (an ideal rational decision vs. targeting a human like decision). Table-1 illustrates the 4 main research flows. According to the "cognitive modeling" stream (i.e. human thinking), AI should be a machine of mind . It contains performing human thinking, based not only on the product itself, as a person when given the same inputs, but on the same logical steps that lead to the same conclusion. “Laws of intellect” stream (i.e. rational thinking) needs an AI to reach to rationalistic decision in spite of what humans may answer.

Table 1-AI research streams based on Russell and Norvig [15].

Objective / *Application to*	*Humanly*	*Rationally*
Thinking	**Cognitive Modeling**	**laws of thought**
Acting	**Turing Test**	**Rational Agent**

Therefore, AI must follow-up the laws of thought through using computational models. Turing Test (i.e. behaving humanly) includes that an AI) should act intelligently when interacting with persons. AI should perform people tasks at minimum good as persons. Those requirements could experiment through the test of ‘Turing, The “Rational Agent” flow considers AI as rationalistic or smart. In this case, the agent does not act independently to achieve a rational ideal result [15].

Intelligent Agent

An IA is an entity that is „capable of ,elasticallly ,separating the .actions to meet the objectives of the design, where elasticity have three implications, namely, reactivity (has a ,response to ,change), pro-activeness (behavior directed towards the target), and ,social ,ability (has the .interaction .with other agents). These properties are requirements to understand the concept of intelligent agents [16].

1.3 Related Works

Several researchers have expressed interest in the intelligent agent, as it has an important role in all aspects of life. Below are some published works related to this topic.

Liu [17] aimed to provide a comprehensive literature review on the use of intelligent agent technology in the library environment. The majority of the literature covered digital libraries (DLs) with fewer studies about services in traditional libraries. The application of agent technology in libraries is still at the experimentation and research stage.

Calvare et al. [18] presented a systematic literature review of studies involving Multi-Agent Systems (MAS) and Block Chain Technologies (BCT) as solutions. The technology of MAS is widely used for the development of intelligent distributed systems (IDS) that manage sensitive data (e.g., healthcare, ambient assisted living, energy trading). Recent trends recommend to use BCT for MAS. The authors aimed to provide a comprehensive overview of their application domains. They analyzed assumptions, motivations, requirements, limitations, and strengths presented in the current time. Moreover, they discussed the future challenges and introduced their vision on how MAS and BCT could be combined in different application scenarios.

Khan and Bhatti [19] aimed to test the use of semantic web technologies for digital libraries. They also investigated the conceptions of university academicians and librarians in Pakistan in relation to semantic web technologies and their uses in digital libraries. An analysis of interview data was performed to obtain results. The findings of this paper showed that semantic information, Dura Cloud, onto edit, and resource description framework (RDF), are the different semantic web applications that can be useful for digital libraries to develop semantic relationships among the digital contents and increase their accessibility in the web environment. The obtained results revealed that semantic web produces precise results and meets the needs of user information in an effective way. They also showed that the next-generation of digital libraries will use context-awareness technology, detecting sensors, and software of intelligent agents to analyze user information needs and provide dynamic services. This paper conceives the future services of digital libraries and the semantic web applications that they can employ.

Miiller et al. [20] proposed an individual logic-based mechanism that amends reliability information to the data shared among the MAS. If multiple agents report the same event, their information is fused. In order to maintain high reliability, the machine detects and isolates misbehaving agents. Therefore, an attacker model is specified that includes faulty as well as malicious agents. The mechanism is applied to Intelligent Transportation Systems (ITS). It was shown in a simulation that the approach scales well with the size of the MAS and that it is able to efficiently detect and isolate misbehaving agents.

1.4Methodologies

WebSocket

The WebSocket protocol is the HTTP(S) based standard that can full- duplex real-time communication, between a web-server and its clients, i.e. typically uniform browser applications [21], as shown in Figure-2.

The Web-Socket protocol is designed to replace the present bi-directional technologies of communication that utilize HTTP as a transported layer to interest from the available infrastructures (authentication, proxies, and filtering). These communication technologies were conducted as a trade-off between efficiency and reliability because, at first, HTTP was not prepared to be used for bi-directional communication technologies.

The WebSocket protocol tries to communicate with the targets of the present bi-directional HTTP communication technologies in the present HTTP infrastructure; it is designed to work through the HTTP executors 80 and 443, as well as to back HTTP [22].

The WebSocket protocol establishes a connection between the server and multiple clients. It is mainly used for real-time data exchange and interaction and for building collaborative applications. One of the most common examples of WebSocket is the chat application. For example, you have many clients accessing the server. When a client sends a message, the server will broadcast the message to all other clients. A more integrated application could be the collaborative role; for example, if there is a class doing e-learning for many students and the teacher is drawing on the web to show how to make a certain shape, all the drawings will be shown in the client-side at the same time. Also, WebSocket is good for stock market trading. It will send the stock price to all clients who can buy or sell at the same time.

WebSocket (WS) server with express is a WS in node.js server. One can initiate the Node Package Manager (NPM) and use it to install the express ws. In this situation, when the WebSocket is installed as a server, then we will have the client to install the reconnecting- html5-WebSocket. This way, the client will be available to access the WebSocket server.

A description of the server is provided below:

```
var express = require('express');
var socket = require('socket.io');
// App setup
var app = express();
var server = app.listen(4000, function(){
   console.log('listening for requests on port 4000,');
});
```

For the client, we have added the following in the web page for the client to access the WebSocket server:

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/1.7.3/socket.io.js"></script>
```

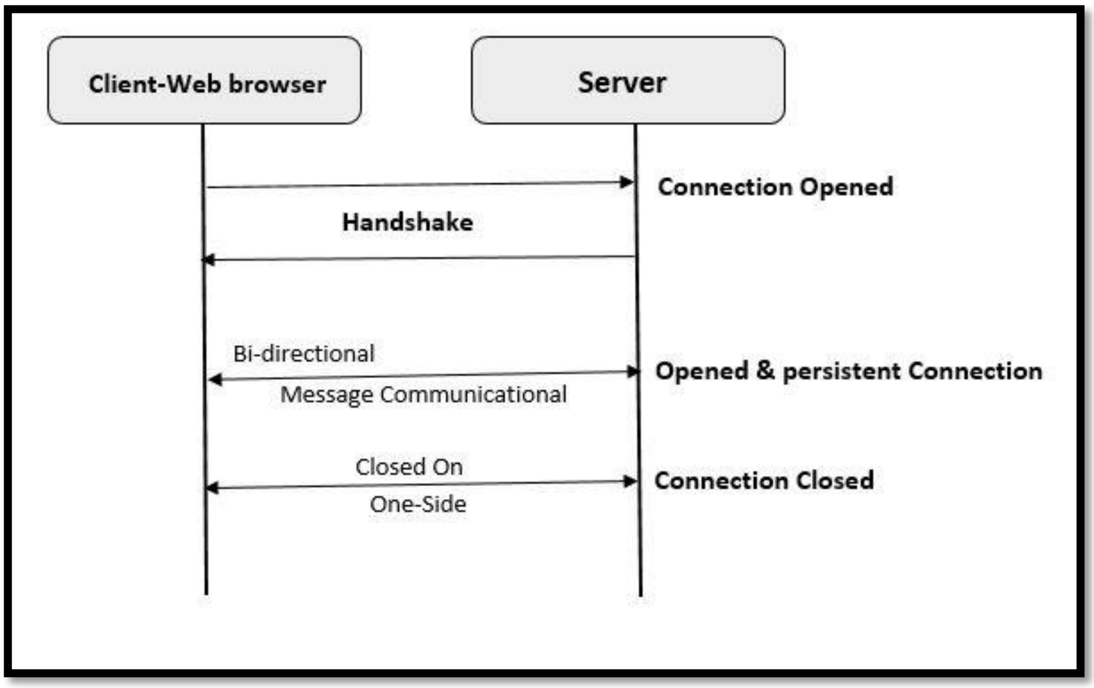


Figure 2-The WebSocket of the used model [21].

Cloud Storage

The usage of Cloud Storage means that a user/customer/company will save their data within the cloud instead of on a local system. Access to the data is consummated by network connectivity and client service. One advantage of Cloud Storage is that customers can access their data from any location, even if they do not have access to their organization's network [23]. Stronger wireless networks and greater use of mobile devices have increased the reliance on online storage [24].

There are many benefits to storing data in the cloud over local storage.There are many benefits to storing data in the cloud over local storage.Companies only pay for the storage they use. It creates operating expenses rather than capital expenses,The data is quickly accessible and reliable. The data is located on the web across multiple storage systems instead of a local site, Better protection in case of a disaster. Sometimes, the organization has a local backup and in cases of fire or natural disaster, the backup will not be available. Cloud vendors provide hardware redundancy and automatic storage failover. This helps to avoid service outages caused by hardware failure. The vendors know how to distribute copies to mitigate any hardware failure.Virtually limitless storage capacities. If the customer does not have the necessity of extra storage, the costs will decrease.Workload balance. Cloud vendors help customers to achieve the best performance by balancing workloads and a unified view of storage. Cloud vendors provide an export to get a unified view of storage utilization [23] [25].

The architecture of cloud computing is defined by layers. There are three main layers; the cloud infrastructure (IaaS), cloud application platform (PaaS), and cloud application software (SaaS) as shown in Figure-3.

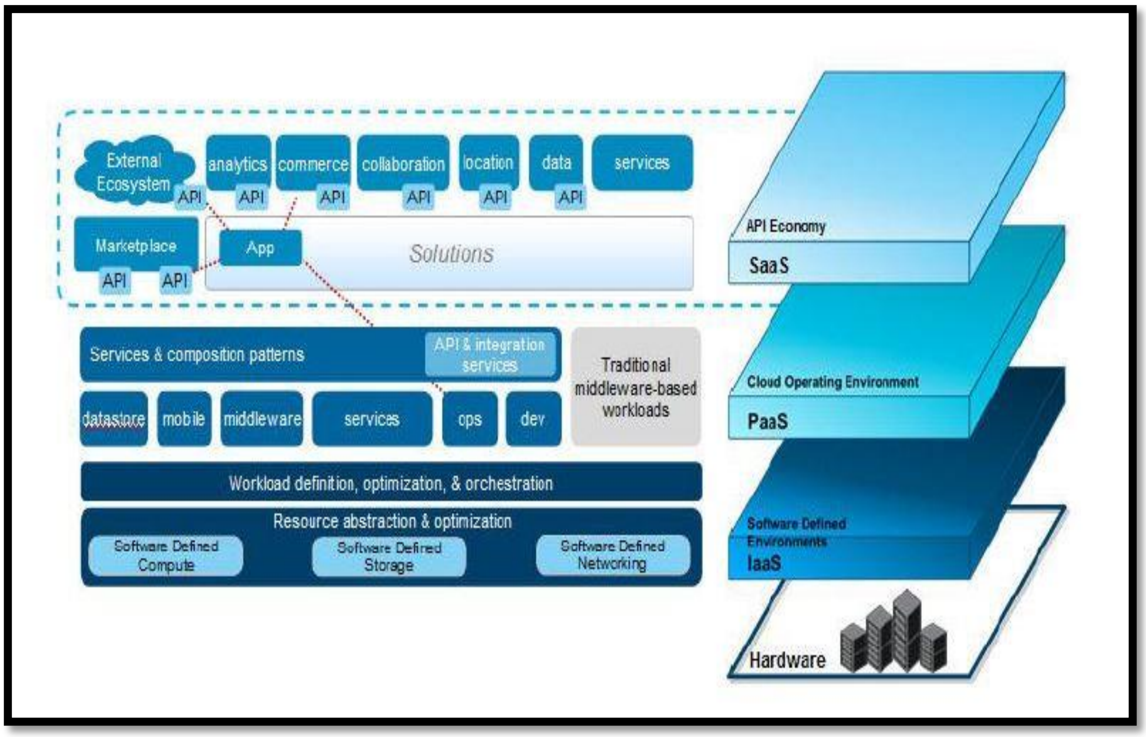

External Ecosystem
API
analytics
commerce
collaboration
location
data
services
Marketplace
App
Solutions
Services & composition patterns
API & integration services
Traditional middleware-based workloads
datastore
mobile
middleware
services
ops
dev
Workload definition, optimizaton, & orchestration
Resource abstraction & optimization
Software Defined Compute
Software Defined Storage
Software Defined Networking
API Economy
SaaS
Cloud Operating Environment
PaaS
Software Defined Environments
IaaS
Hardware

Figure 3-The Architecture of Cloud Computing [26].

The layer that includes cloud storage is Infrastructure as a Service (IaaS). Physical and virtual resources used to construct the cloud are included in this layer.
Resources are provided and managed in physical and virtual servers. Storage pools are unaware of what applications are running on them [27].

2. Materials and Methods

The architecture of the proposed model

The proposed model consists of five parts (users or client, www, proxy server, websocket, and agent) as shown in Figure-4.

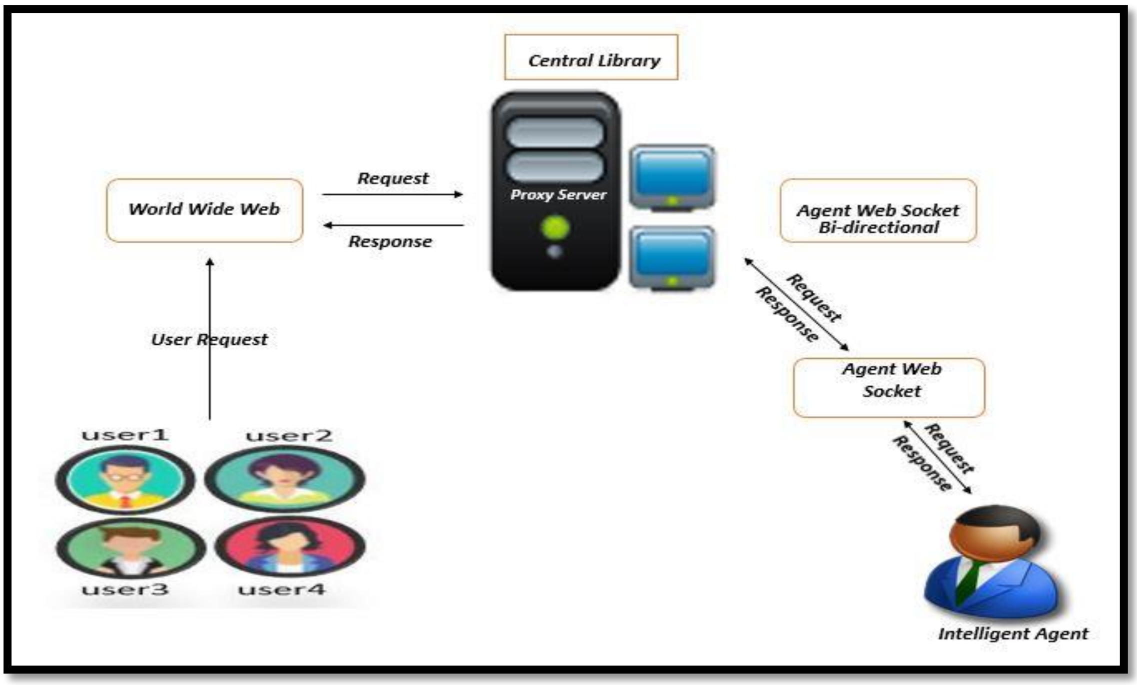
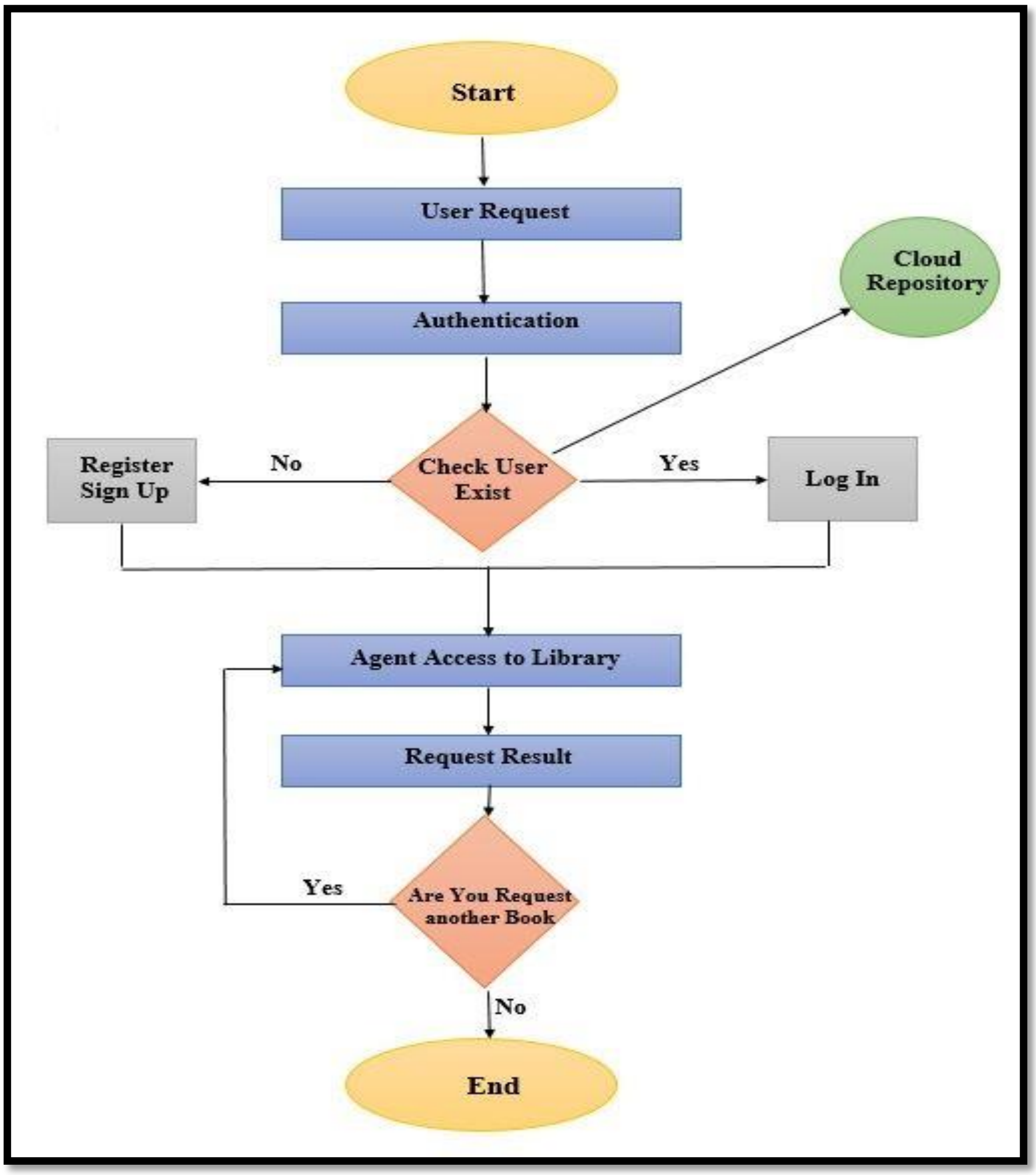


Figure 4- The proposed model of library architecture.

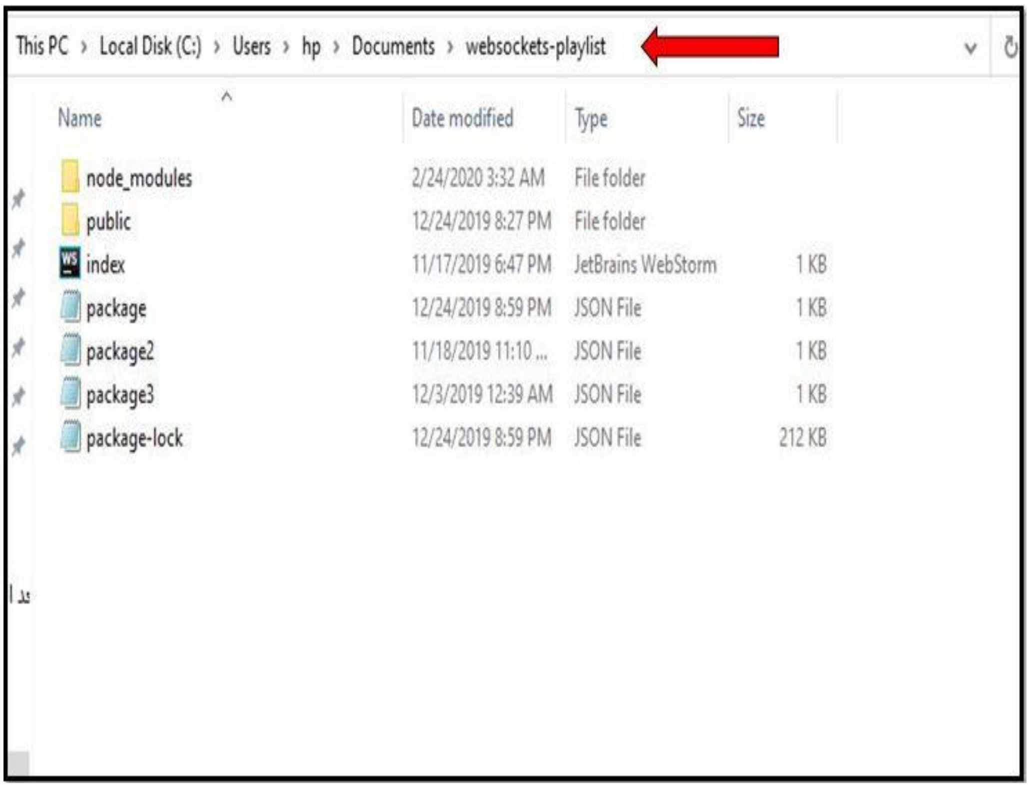
The proposed model includes designing a fast program for a library, where every user can achieve fast results. The flowchart of the proposed model as shown in Figure-5.



Finger 5- Flowchart of the proposed model .

3. Results and discussion

The proposed model is implemented in JavaScript programming, Firebase database, WebSocket Agent, and Node Js, using a personal computer (laptop) The experiments were performed on an Intel (R) Core (TM) i7-8565U CPU @ 1.80 GHz 1.99 GHz, 64 bit Operating System, and 8 GB RAM. The model is very fast, with measures can be accomplished in milliseconds according to the size of the server. Implementation results for each step are shown below to complete the proposed form.
Copies for the path of the WebSocket playlist were produced, as shown in Figure-6.

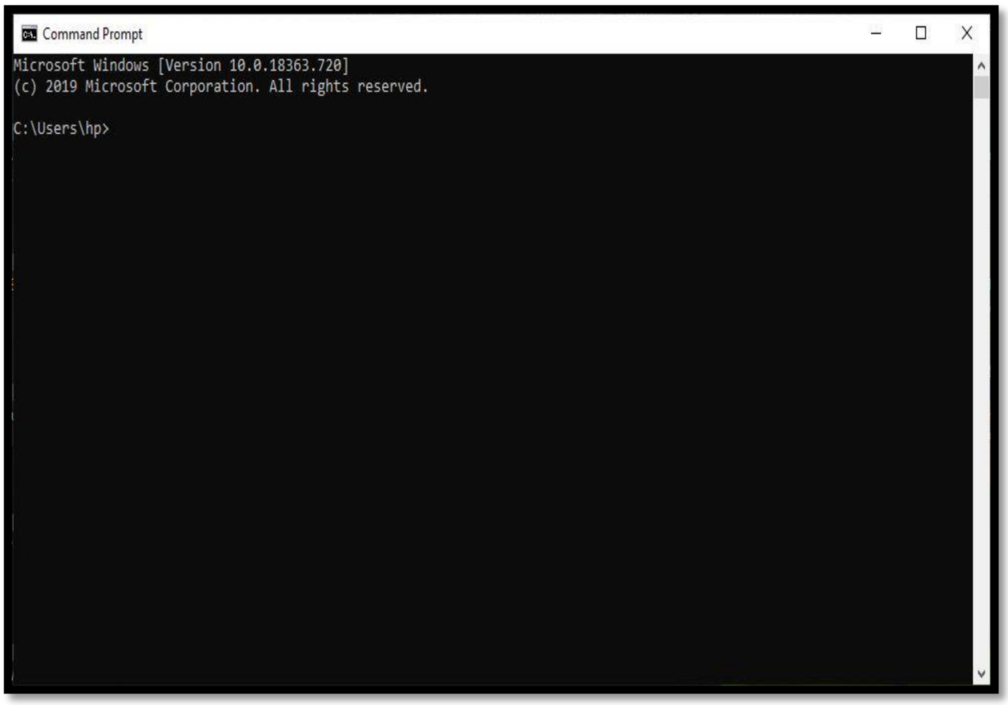


```
This PC › Local Disk (C:) › Users › hp › Documents › websockets-playlist

Name              Date modified        Type                 Size
node_modules      2/24/2020 3:32 AM    File folder
public            12/24/2019 8:27 PM   File folder
index             11/17/2019 6:47 PM   JetBrains WebStorm   1 KB
package           12/24/2019 8:59 PM   JSON File            1 KB
package2          11/18/2019 11:10 ... JSON File            1 KB
package3          12/3/2019 12:39 AM   JSON File            1 KB
package-lock      12/24/2019 8:59 PM   JSON File            212 KB
```

Figure 6- The path of the program

Then, the Command Prompt, an executable file which is named cmd.exe, was opened, that is a command- line interpreter available on Microsoft, as shown in Figure-7.

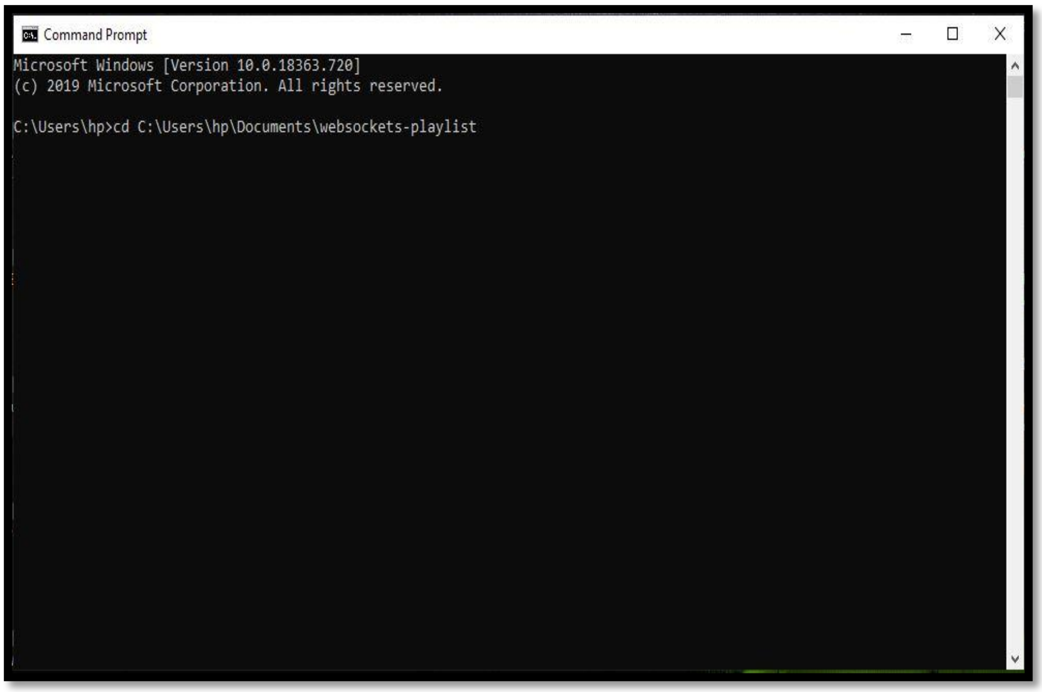


```
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>
```

Figure 7- The Command Prompt

After that, command “cd” was written, which is a shortcut to the change directory. Then the path above was copied and pasted, as shown in Figure-8.

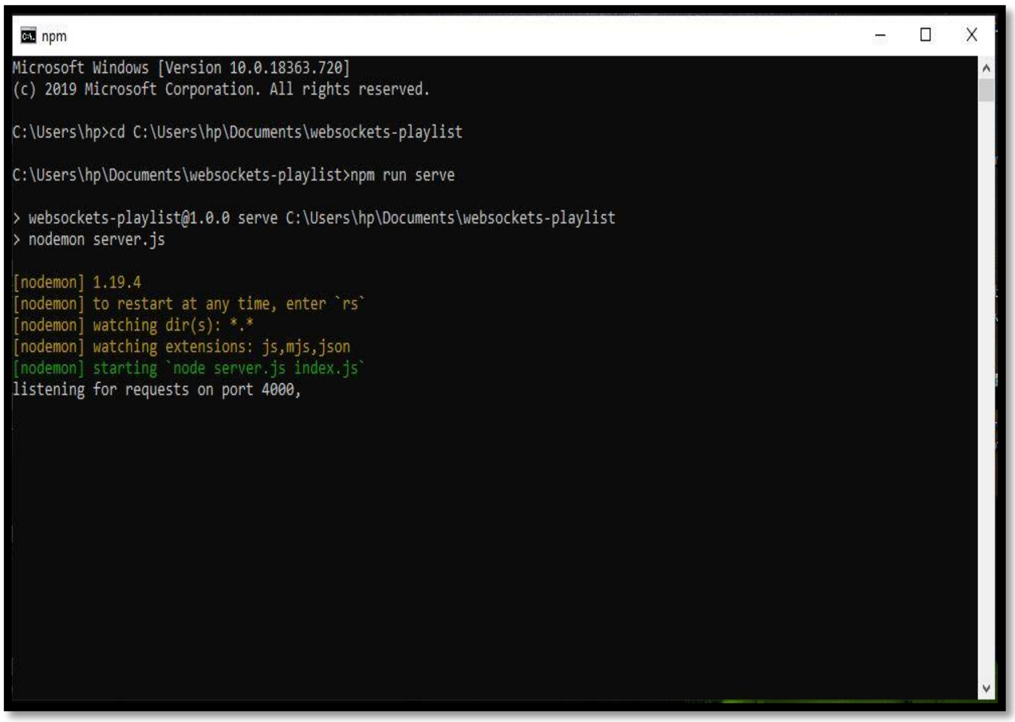


```
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\Documents\websockets-playlist
```

Figure 8- Running the command "CD" with the path of webSockets-playlist

After that, the command "npm run serve" was run, where the program will run, as shown in Figure-9.



```
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\Documents\websockets-playlist

C:\Users\hp\Documents\websockets-playlist>npm run serve

> websockets-playlist@1.0.0 serve C:\Users\hp\Documents\websockets-playlist
> nodemon server.js

[nodemon] 1.19.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server.js index.js`
listening for requests on port 4000,
```

Figure 9-Running the Command "npm run serve"

When using the model, firstly it should run the server. After that, any browser can be opened where the following link is entered: http: // localhost: 4000 / agentAuth.html. Then "Enter" is clicked and the form homepage opens, as demonstrated in Figure-10.

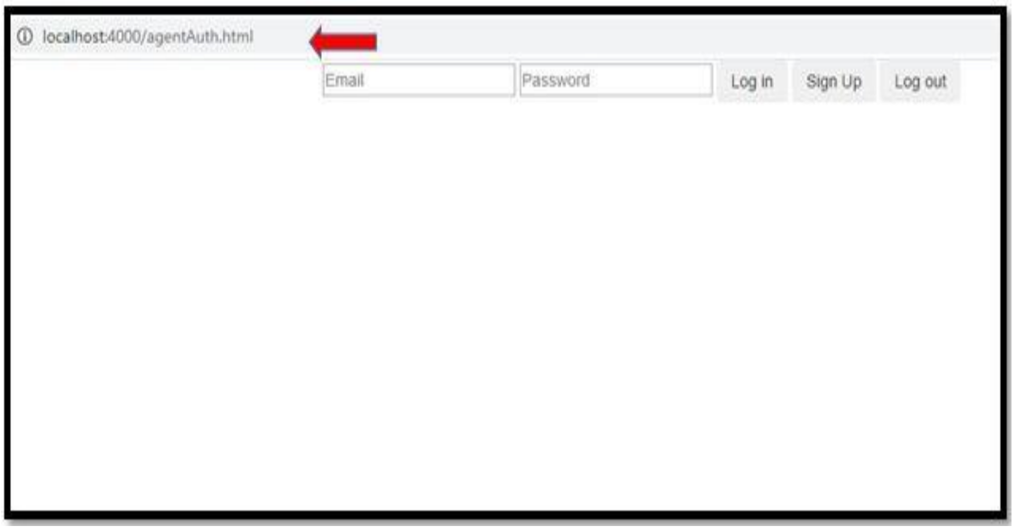


Figure 10-Home Page of AgentAuth

When the users or clients want to login to the electronic library, they enter the email and password and click on the login as shown in Figure-11.

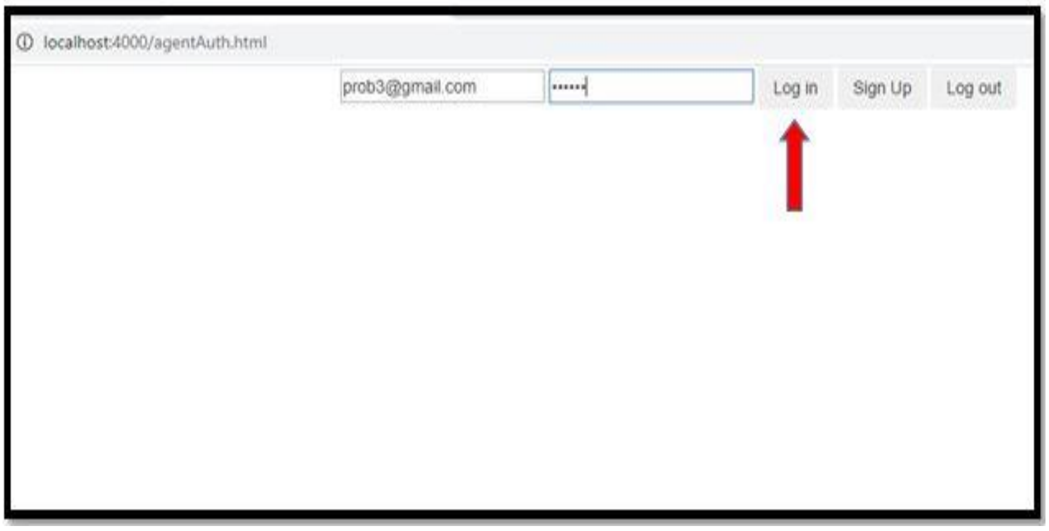


Figure 11-The login process.

If not registered to the model, the user will press sign up, as shown in Figure-12.

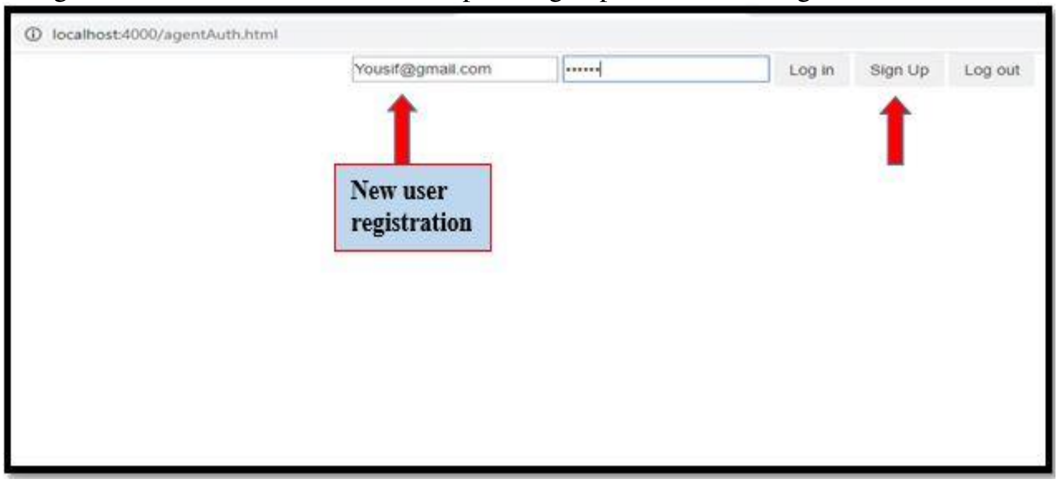


Figure 12-The Registry Process for a new user

As shown in Figure-13, a connection is made between the browser and the server, which means that the agent has entered based on the WebSocket. Any user who connects to the model will notice all the communications because it is open communication to everyone.

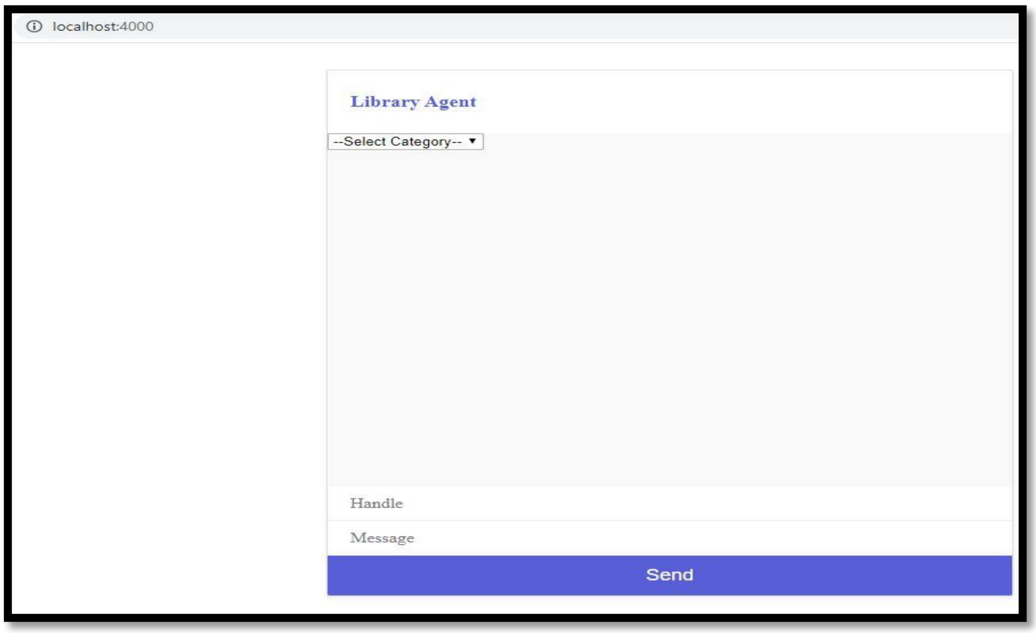
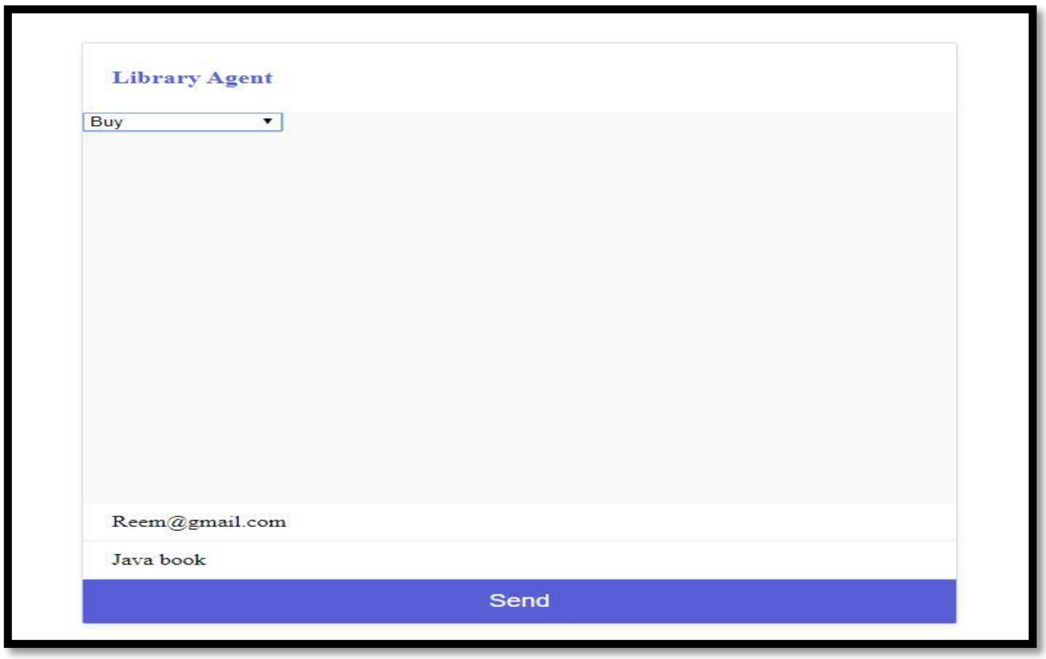


Figure 13-Connect and entered the library

After entering the library, they users should select the desired category. There are three categories: buy, rent, and borrow. After selecting the category, the email address should be written in the field (handle) and the name of the book in the field (message), then a click on the "send" bottom should be made, as shown in Figure-14.


Library Agent

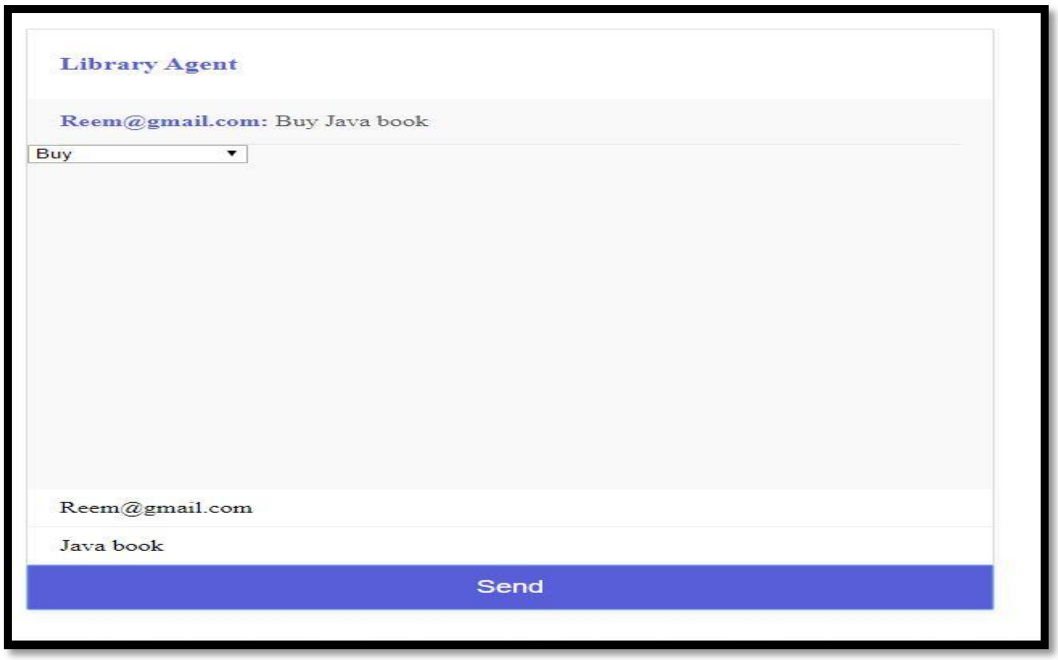
Buy

Reem@gmail.com

Java book

Figure 14-The process of ordering a book.

The next step of the asynchronization process is sending the request by the user, as shown in Figure-15.


Library Agent

Reem@gmail.com: Buy Java book

Buy

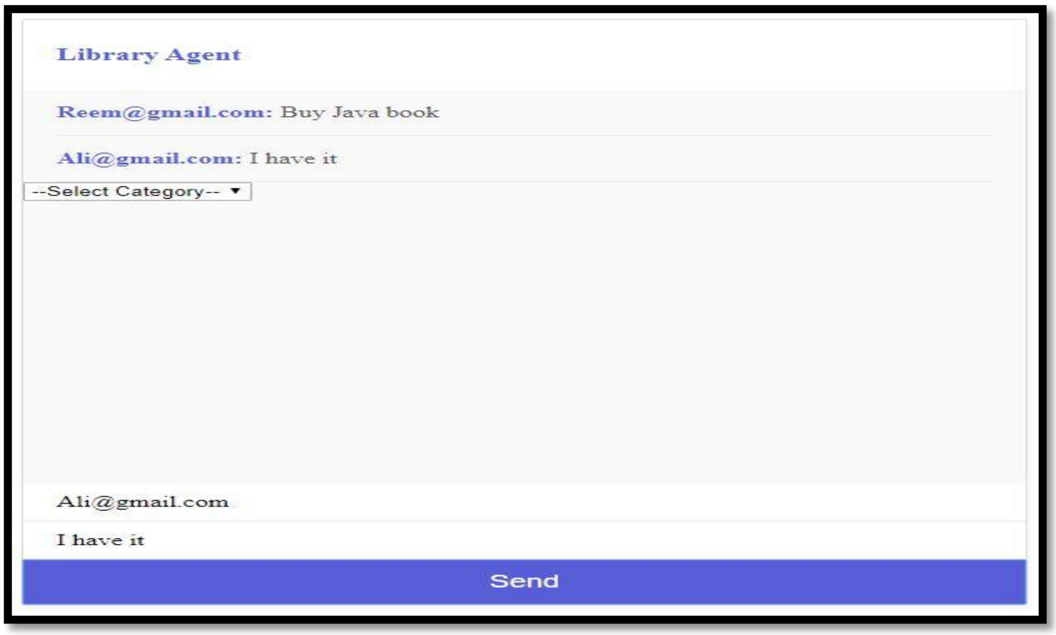
Reem@gmail.com

Java book

Send

Figure 15-The second step of asynchronization process.

In the third step of the asynchronization process, the second user or agent responds to the request of the first user, as shown in Figure-16.


Library Agent

Reem@gmail.com: Buy Java book

Ali@gmail.com: I have it

--Select Category--

Ali@gmail.com

I have it

Send

Figure 16-The Third step of asynchronization process

When the first user sends the request and the second user's responds that the book is in his/her possession, the second user sends the book to the email of the first user.

4.1 Visual comparison between HTTP and WebSocket proposed model:

In this section, a comparison between the proposed model that uses the WebSocket and that of the Web server is made based on main features. Table-2 Illustrates the results of this comparison.

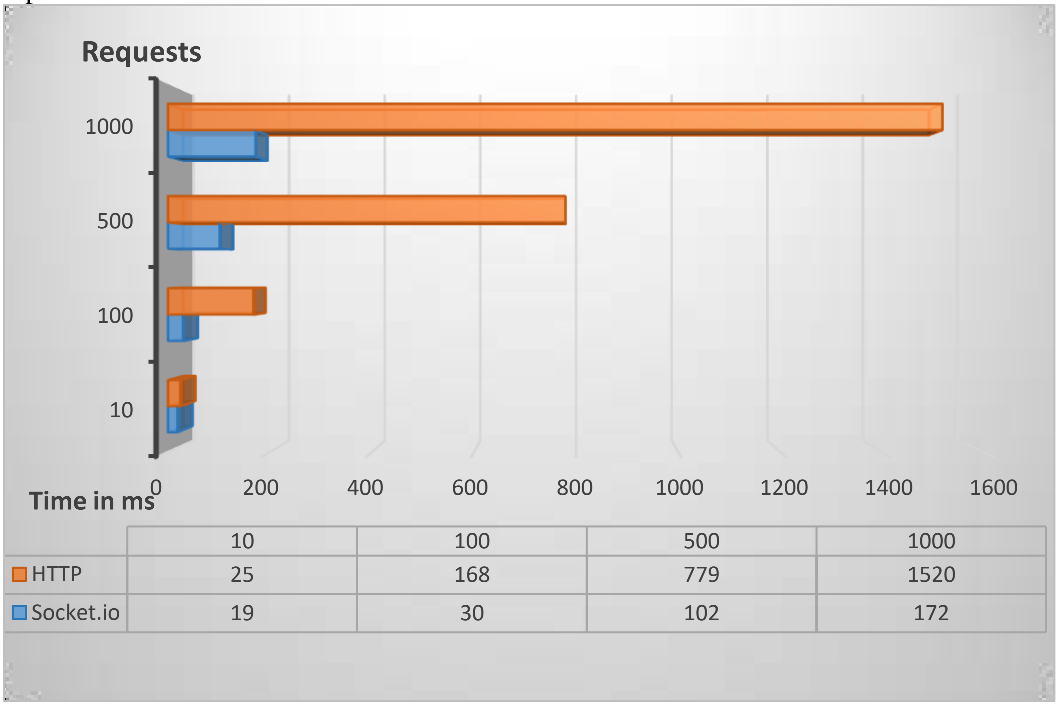
Table 2-Comparison between the proposed model that uses the web socket and the web server.

The proposed WebSocket model	Web Server
Duplex	
Full duplex	Half duplex
Messaging pattern	
Bi-direction	Request-Response
Services push	
Core feature	Not natively supported. Client polling or streaming download technique used.
Supported Clients	
Modren languages and Clients	Broad support
Overhead	
Moderate overhead to establish and maintain the connection, then minimal overhead per message.	Moderate overhead per request / connection.
Intermediary / Eadge caching	
Not possible	Core feature

4.2 The proposed model efficiency

The time required to implement the processes with the proposed model is very fast, being measured in millisecond, depending on the server being used and the speed of the network. If the server is very large, then the processes will be implemented in nanoseconds.

. On average, the time required to process 10 requests using HTTP was about 25 ms, while it was 19 ms when Socket.io was utilized for the processing of 100, 500, and 1000 requests, time values were 168 ms, 779 ms, and 1520 ms using HHTP and 30 ms, 102 ms, and 172 ms using Socket.io, respectively. As is clear from Figure-1 , for the purposes targeted in this work, WebSocket is about 5-7 times faster than the plain HTTP.



	10	100	500	1000
🟧 HTTP	25	168	779	1520
🟦 Socket.io	19	30	102	172

Figure 17-The proposed WebSockets model. The results show that the model is about 5 to 7 times faster than the plain HTTP.

Figure-18 demonstrates a Cartesian chart of the results of the efficiency of processing the requests achieved by the proposed model.

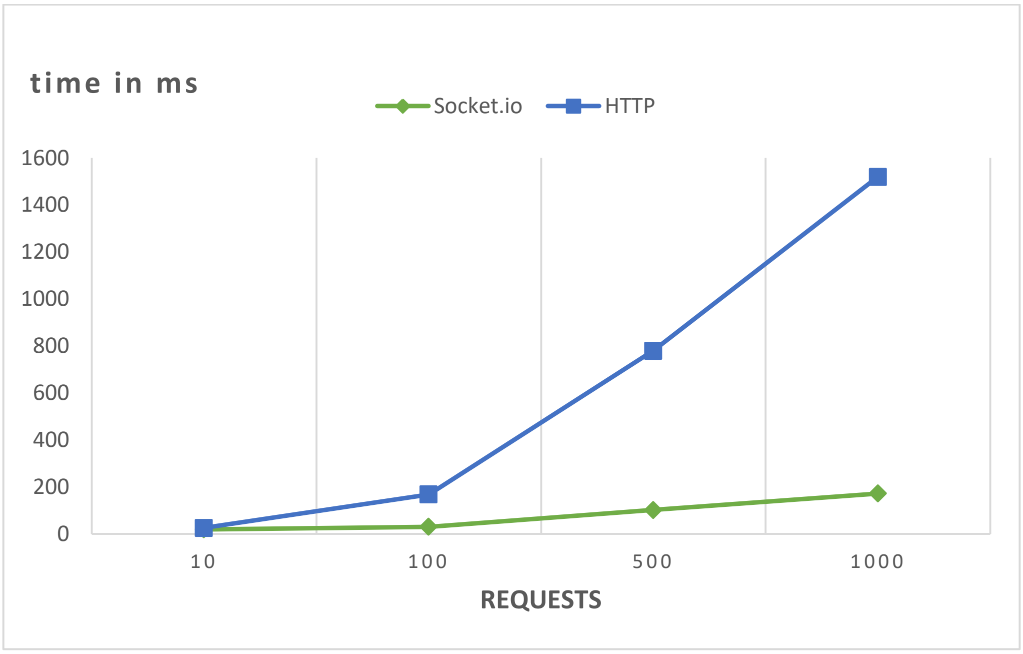


Figure 18- A Cartesian chart of the efficiency of precessing the requests achieved by the proposed model.

4.3 Problems and difficulties

The problems that were encountered during this work were the installation of the node.js and the selection of the proper client WebSocket package.

Conclusions

Agent and multi-agent techniques are considered as modern technologies that deal with distributed systems, mainly web and database applications. Agent services and web application in a distributed system are more complicated than those in a regular system. In the present work, an intelligent agent was used in an e-library model. A firebase was used as a database where a fast and continuous communication process was achieved until the browser was closed by the user, the computer is turned off, or the server is off. WebSocket in_the e-libraries was not used before in Iraq, Arabian countries, or Iran. The model works in a multi-environments (UNIX, Windows, Mac, or any other operating system). The WebSocket agent used in the present work also supports different browsers (Chrome, Firefox, and Internet Explorer). It also supports all languages.

References

1. Green, S., Hurst, L., Nangle, B., Cunningham, P., Somers, F., and Evans, R. **1997**. *Software agents*: A review. *Department of Computer Science, Trinity College Dublin, Tech. Rep. TCS-CS-1997-06*.
2. Russel, S., and Norvig, P. **1995**. *Artificial Intelligence: A Modern Approach*. *Cited on*, *20*, 90020-9. Jennings, N., Jennings, N. R., and Wooldridge, M. J. (Eds.). 1998. *Agent technology: foundations, applications, and markets*. Springer Science & Business Media, E-Book, 11-17.
3. Roy, S., Banerjee, A., Ghosh, P., Chatterjee, A., and Sen, S. **2019**. Intelligent web service searching using inverted index. In Contemporary Advances in Innovative and Applicable Information Technology (pp. 13-21). Springer, Singapore.
4. El Yamany, H. F., Capretz, M. A., and Capretz, L. F. **2006**. A multi-agent framework for testing distributed systems. In 30th Annual International Computer Software and Applications Conference *(COMPSAC'06)* (**2**: 151-156). IEEE.
5. Sommerville, I. **2004**. *Software processes*. *Software Engineering*, (PART. 1).
6. https://www.cs.auckland.ac.nz/compsci230s1c/lectures/xinfeng/process.pdf
7. Tsai, W. T., Yu, L., Saimi, A., and Paul, R. **2003**. Scenario-based object-oriented test frameworks for testing distributed systems. In The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems, 288-294.

8. Long, B., and Strooper, P. **2001**. A case study in testing distributed systems. In Proceedings 3rd International Symposium on Distributed Objects and Applications, 20-29.

9. Lastovetsky, A. **2005**. *Parallel testing of distributed software*. Information and Software Technology, 657-662.

10. Schommer, C. **2008**. An unified definition of data mining. *arXiv preprint arXiv:0809.2696*.

11. I. H. Witten, E. Frank, and M. a. Hall. **2011**. *Data Mining: Practical Machine Learning Tools and Techniques*, Third Edition.

12. Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., and Kingsbury, B. **2012**. Deep neural networks for acoustic modeling in speech recognition: *IEEE Signal processing magazine*, **29**(6): 82-97.

13. He, K., Zhang, X., Ren, S., and Sun, J. **2016**. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition 770-778.

14. Koza, J. R., Bennett, F. H., Andre, D., and Keane, M. A. **1996**. Automated design of both the topology and sizing of analog electrical circuits using genetic programming. In *Artificial Intelligence in Design'96*, Springer, Dordrecht, 151-170.

15. Russel, S., and Norvig, P. **2015**. Artificial intelligence: a modern approach, 3rd Education, Pearson.

16. Šlapák,M., 2019."Adaptive Control Algorithm of Intelligent Agents", PhD. Thesis, the Faculty of Information Technology, Czech Technical University in Prague.

17. Liu, G. **2011**. The application of intelligent agents in libraries: a survey. *Program*.

18. Calvaresi, D., Dubovitskaya, A., Calbimonte, J. P., Taveter, K., and Schumacher, M. **2018**. Multi-agent systems and blockchain: Results from a systematic literature reviewIn International conference on practical applications of agents and multi-agent systems , Springer, Cham, 110-126.

19. Khan, S. A., & Bhatti, R. 2018. Semantic Web and ontology-based applications for digital libraries. *E- Library*.

20. Müller, J., Meuser, T., Steinmetz, R., and Buchholz, M. **2019**. A Trust Management and Misbehaviour Detection Mechanism for Multi-Agent Systems and its Application to Intelligent Transportation Systems. In IEEE 15th International Conference on Control and Automation (ICCA*)*, 325-331.

21. Imre, G., and Mezei, G. **2016**. Introduction to a WebSocket benchmarking infrastructure. In Zooming innovation in consumer electronics international conference (ZINC), 84-87.

22. Fette, I., and Melnikov, A. **2011**. The websocket protocol.

23. Galloway, J. M. **2013**. A cloud architecture for reducing costs in local parallel and distributed virtualized cloud environments. Ph.D. Thesis,University of Alabama Libraries.

24. Erturk, E., and Iles, H. R. E. **2015**. Case study on cloud based library software as a service: Evaluating EZproxy. *arXiv preprint arXiv:1511.07578.*

25. Obrutsky, S. **2016**. Cloud storage: Advantages, disadvantages and enterprise solutions for business. In *Conference: EIT New Zealand.*

26. Diaz, A., and Ferris, C. **2013**. Ibm's open cloud architecture. *IBM Corp., Armonk, New York*.

27. Marks, E. A., and Lozano, B. **2010**. *Executive's guide to cloud computing*. John Wiley and Sons.