International Journal of Bio-Science and Bio-Technology
Vol. 8, No.3 (2016), pp. 239-248
http://dx.doi.org/10.14257/ijbsbt.2016.8.3.24

Agent Based Yoga Recommendation System for Better Health

Abhishek Mathur[1], Siva Shanmugam.G[2] and N. Ch. S. N. Iyengar[3]

School of Computing Science and Engineering,
VIT University, Vellore – 632014, Tamil Nadu, India
abhishek.mathur2013@vit.ac.in[1],sivashanmugam.g@vit.ac.in[2],
nchsniyr@vit.ac.in[3]

Abstract

This work is an effort to implement an agent based system that can recommend the authentic yoga asana to be done by an user (patient) taking into due consideration of chronic health problems, the current symptoms, age factor and diet. Although there are many websites on yoga today and a lot of information is available over the internet but this system would first analyze the problems of his user and then give its advice which will be precise, accurate and authentic. The agent will also tell the user to take appropriate precautions based on her age and diet if applicable.

Keywords*: yoga, asana, agent technology, mobile agents, JADE, FIPA.*

1. Introduction

In the present scenario where internet is filled with huge amount of resources on doing exercises , meditation and other therapies this work comes as a customizing tool for the user which enables her to get direct answers from the artificial agent instead of searching online and talking to different people and hence being misguided many a times. It is important to realize that yoga like other exercises gives benefits when done in a proper way and might cause harm or even injuries when done incorrectly. Hence it is important that our agent has adequate knowledgebase so that it can give warnings and suggestions to avoid any kind of physical stress or injury. Coming to the implementation part of this project following is a comparative study of different technologies that can be used to make such an intelligent application [1].

Table 1. A Comparative Analysis of Agent Development Technologies

Platform Features	Aglets	JADE	Voyager	TACOMA	Grasshopper	SPRINGS
Organization	IBM Tokyo	Research Telecom Italia Lab	Object Space	Tromoso& Cornel University	IKV++	Distributed Information Systems Group
Model	Events	Behaviors	Procedural	Procedural	Procedural	Procedural
Compliancy to agent Standard	MASIF	FIPA	---	---	FIPA	---
Programming languages	JAVA	JAVA	JAVA	C,Perl, Unix,Tcl scripting language	JAVA	JAVA
Elements	Contexts -Agents	Containers -Main	-Servers -Agents	-DATA -CODE	-Places -Regions	-Places -Regions

ISSN: 2233-7849 IJBSBT

	(aglets) -Tahiti	container -Platforms Agents		-HOST -meet	-Agents	(RNSs) -Agents
Communication Technique	Synchronous Asynchronous	Asynchronous	All methods	Asynchronous	Synchronous	Synchronous, Asynchronous
Messages	Yes	Yes(FIPA)	No	Yes	Yes(FIPA)	Yes
Mobility	Aglet transfer, Inbuilt agent, Java object, Transfer Dynamic.	-	-	-	-	-
Service	Protocol	Mobility service	series	Control Protocol	Proxies	proxies (location wise)
Available for download	IBM Public license	LGPL	Not free(evaluative version)		Not anymore	Yes (binaries)
GUI Based tools	Some	Yes	No	some	Yes	No
Security	Limited	Strong	Limited secured channel	User firewall agent	Limited	Limited

2. Related Work

Agent based healthcare applications and telemedicine applications have been a great area of research and a lot of progress has been made in this field. A fully functional ABHCS has been designed in [1] using JADE that improvises the traditional way of delivery of medical service. This application connects doctors to patients via a set of agents that work in a synchronized manner in a multi-agent environment. The agent might recommend the drug and tests to the patients based on her symptoms and severity of disease.

In [2] a MADIP system has been proposed and implemented using mobile agents that can work as another framework on top of JADE. It connects one doctor to many patients and comprises of user agents, physician agents and diagnostic agents. The application is distributed over several containers and thus acts as a mobile service. AgentBuilder [3] is a tool for building Java agent systems based on two components: the Toolkit and the Run-Time System. The Toolkit includes tools for managing the agent software development process, analyzing the domain of agent operations, defining, implementing and testing agent software. The Run-Time System provides an agent engine, that is, an interpreter, used as execution environment of agent software. Agents usually communicate through KQML messages; however, the developer has the possibility to define new communication commands to cope with her/his particular needs. dMARS [4] is an agent-oriented development and implementation environment for building distributed system based on the BDI agent model offering support for system configuration, design, maintenance and reengineering. Such a development environment has been successfully used to realize application in the fields of air traffic control and of telecommunication and business process management.

3. Architecture and their Agent Functions.

This project has been coded on Netbeans IDE and oracle JDBC driver using java programming language and JADE. Primary agent is the agent class responsible for overall query filtering and input output operations; logic engine agent performs the mapping of symptoms and data entries which is the heart of the application. As the application uses ORJDBC to resolve the address of the database dynamically we don't have to worry if the database is located on a different host machine or different network or a cloud. All agents resolve the database address at runtime as and when needed. Now coming to the flow of the application, in the first window of the application the user is prompted to enter her name , age , chronic health problems that can be used by the agent during filtering the of recommendations and details of diet taken. In the second step the agent asks the current symptoms for which the user has turned up to us three iterations on the result set. First selecting the rows of result set that match with current symptoms. Second, select those asanas that are recommended at the given age of the patient and in the final iteration select those

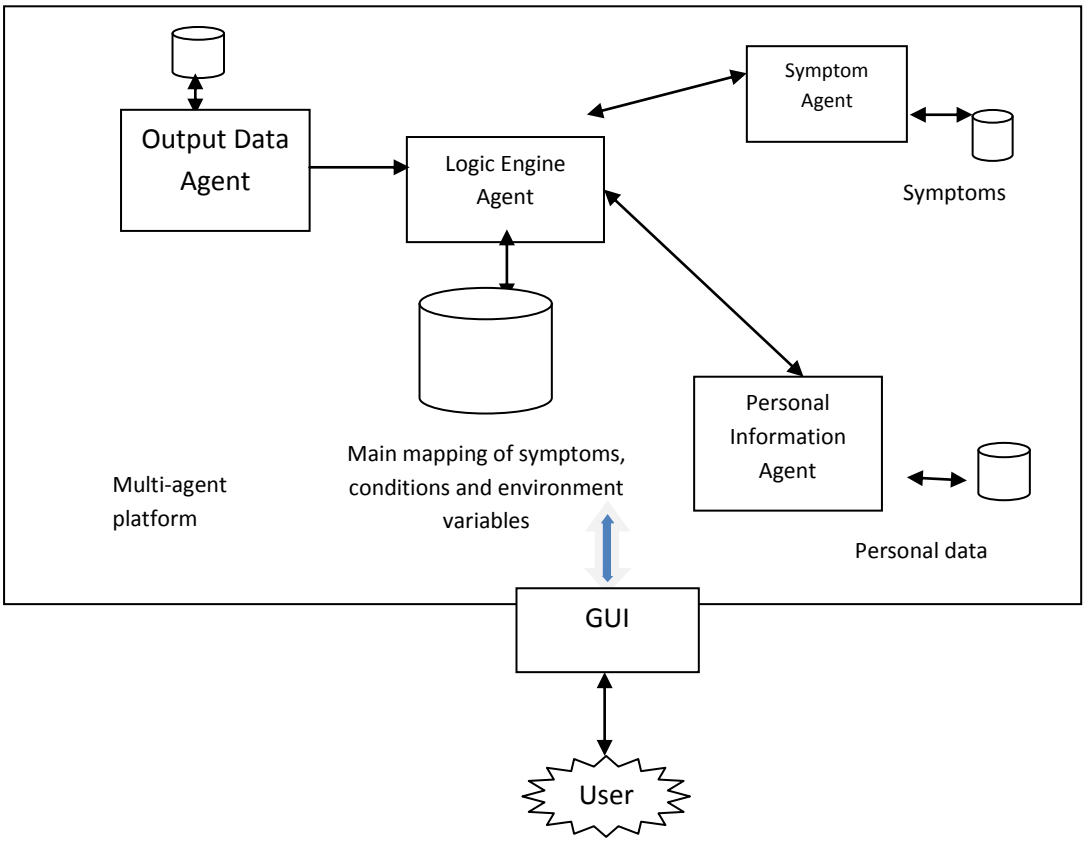

Figure 3. Architecture Diagram

that can be done by people having the given chronic health condition. Lastly generate the diet recommendations corresponding to each asana. We now have the final yoga asana recommendations with the primary agent. It will then fetch the pictures and videos of all the asanas from the database and output it to the user. The individual functionalities of each agent is described as under, all of these agents (java classes) have a GUI window such as a jFrame associated with them. All communication to the user is done using the swing GUI window.

Table 2. Representation of Overall Communication Process

Personal Input Agent (P_Agent)	1. Take the input from its GUI 2. Authenticate the user 3. Connect to its local database. 4. Set the flags and global variables accordingly. 5.Update the its local database with the values given by the user 6. Make these flags and variables available to the logic engine agent when prompted 7. Wait for the user to terminate the application otherwise restart.
Symptoms Agent (S_Agent)	1. Initialize its GUI 2. Give a list of possible symptoms to the user 3. Listen to the input of the user 4. Store these input symbols along with the chronic symptoms entered during personal Input in its database. 5.Give details to logic agent when asked.
Logic Engine Agent	1. Access the databases of P_Agent and S_Agent. 2. Generate a SQL query based on the chronic symptoms and the current symptoms that returns a larger set of data that needs to be filtered. 3. Apply the logic function containing the matrix of nested if-else conditions on this larger data set present in the result set class. 4. Filter out the data that has to be avoided by persons having the given disease. 5. Filter out the data that is not recommended at the given age. 6. Generate the diet advice based on the obtained data till step 5. 7. Sort the asanas that can be done before food and those that can be done after taking meal. 8. Generate advise based on special conditions (like pregnant ladies, persons with trauma etc). 9. Update the result set and send it to the output agent for sequential data retrieval. 10. Wait for error handling or program termination
	11. Save the data in case of closure of JVM suddenly.
Output Agent	1. Initialize its GUI. 2. Receive the result set from the logic engine agent 3. Output the data to the user 4. Close the GUI on exit.

4. Implementation

The implementation part carries with two steps. First P_Agent should be created and second S_Agent should be created. The algorithms for both the agent is shown below with their GUI representations.

4.1. Algorithm for Personal input Agent (P_Agent)

Input: personal data variables representing (name, age, gender, diet and chronic health details)
Output: data variables to logic Agent(L_Agent)

Start

1. Create a class extending jade.boot by including jade jar files in the project.
2. Make a jFrame that acts a GUI for this class
3. Write the setup () method to initialize behaviors of agent.

4. do following as a simple one-shot behavior:
   Connect to database by resolving the address using your JDBC driver
   Take the input in GUI from user
   Put data in local data structures or variable sets
   When finished==true; break
5. Call the takedown () method that will free the allocated memory.

End

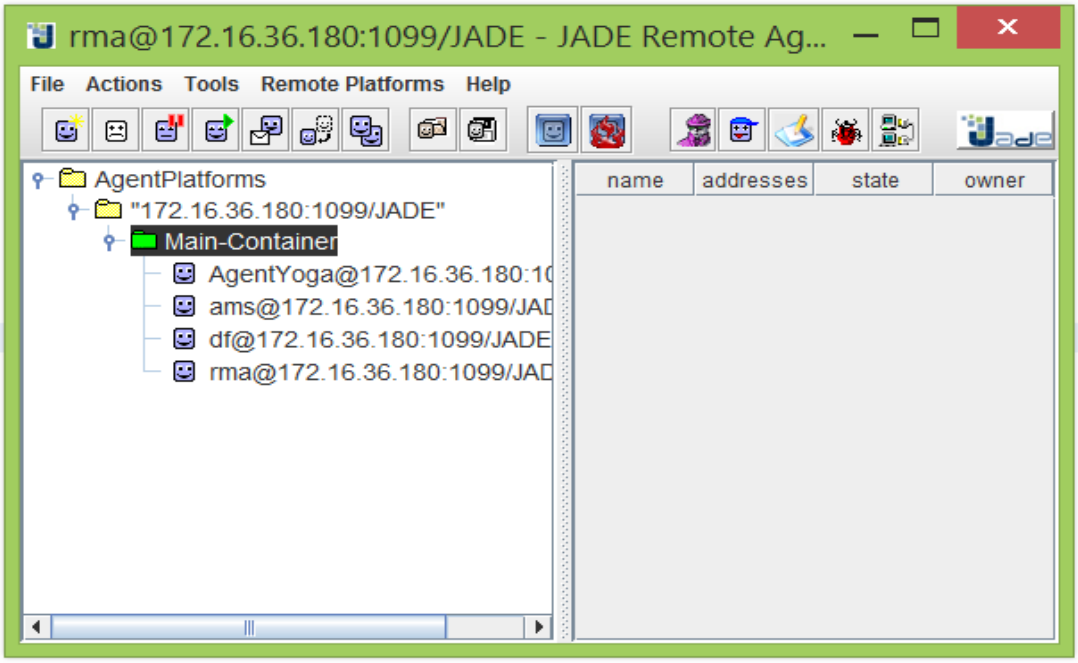

Figure 4.1. JADE Remote Agent Management

4.2. Algorithm for symptom Agent (S_Agent)

Input: symptoms variables fromjList
Output: data variables to logic Agent (L_Agent)
Start
1. Perform steps one to three as in P_Agent
2. In a simple behaviour do the following:
Connect to database
Display all the symptoms as a jList
Allow multiple list selections
Store user selection in local database
3. Wait in while (1) loop till instructed by logic engine agent (L_Agent)

End

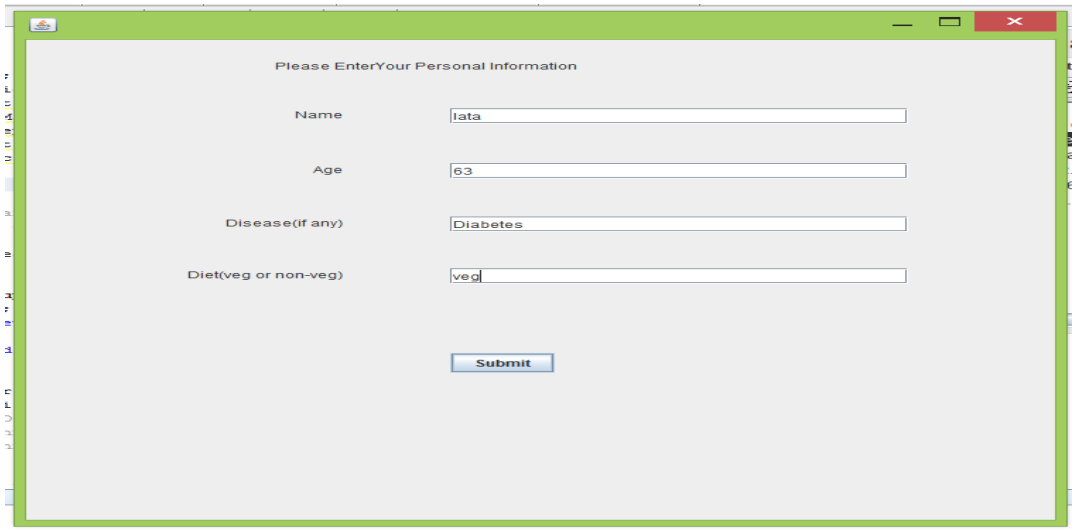

Figure 4.1(a). GUI of P_Agent

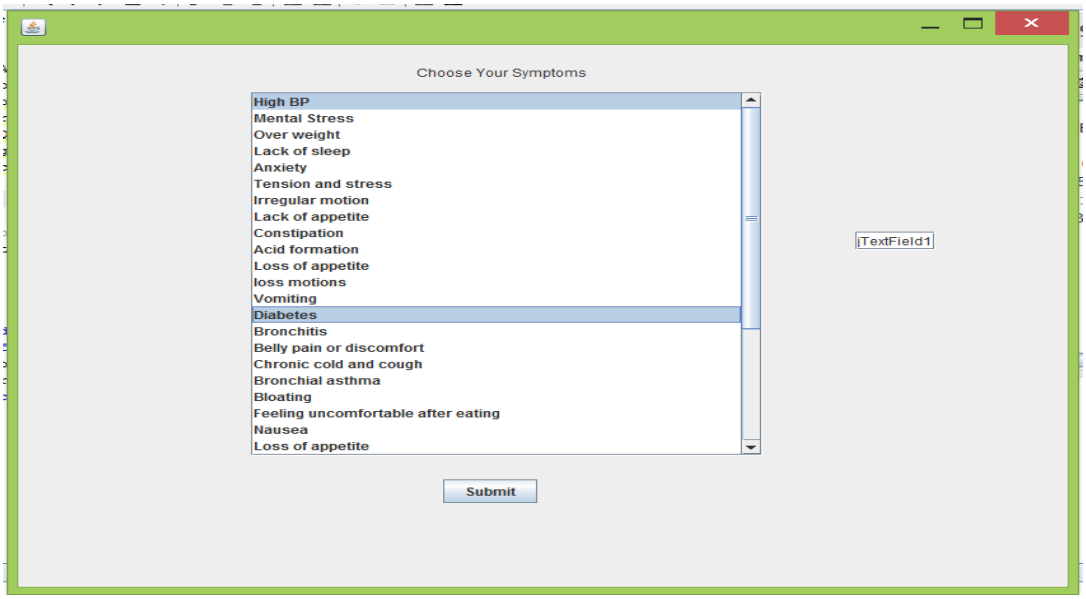

Figure 4.2. jList Framework

4.3. Algorithm for Logic Engine Agent(L_Agent)

Input: data variables from P_Agent and L_Agent
Output: dynamic SQL query in the form of prepared statements, unfiltered result set.
Start
1. Create a result set object (import java.sql.ResultSet)
2. Create a Prepared Statement object which is a directly executable SQL query.
3. Connect to databases of P_Agent and S_Agent and perform join operation based on disease column.
4. Call the logic () function to find the query variables from the matrix of nested if-else conditions.
5. After applying if conditions on all environment variables,
   Select the larger dataset that will be filtered
6. Filter the ResultSet based on chronic diseases, then age and finally generates the diet advice

7. Pass the ResultSet hence obtained to output GUI through output agent using the constructor of that class containing its behaviors.
End

4.4. Algorithm for Output Agent

Input: complete result set from L_Agent.
Output: General advice, steps of yoga asana, name of asana, diet advise, step-by-step photos of each asana.

Start
1. Take result set object from logic engine
2. Retrieve and display all the selected records in corresponding JTextAreas.
3. Set finished == true on clicking finish button.
End

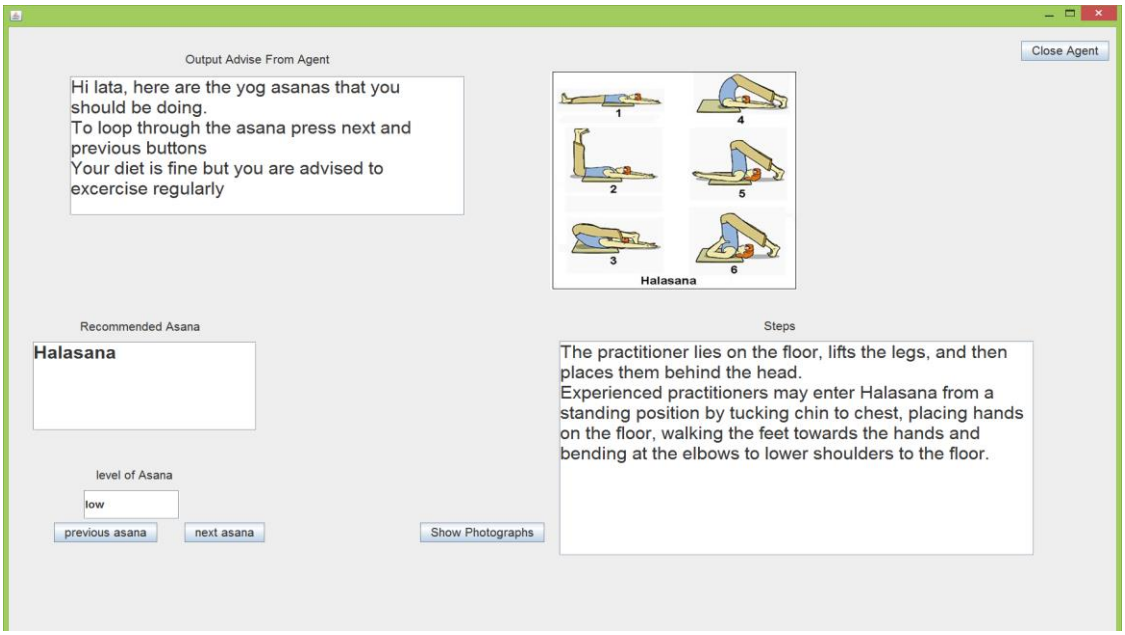

Figure 4.3. Output Advise for different Asanas

5. Performance Analysis

The agent system was tested by the help of yoga instructors who are doing yoga with Bhartiya Yoga Sansthan for the last 8-10 years. The feedback received from them was that this system efficiently recommends yoga asanas based on many factors and can be helpful for beginners in getting started. But yoga is not just about doing asanas, it is rather a way of life. It has eight components known as AshtangYogas- Yam, Niyam, Asana, Pranayama, Pratyahar, Dharna, Dhyan and Samadhi. Each of these together makes a person a complete yoga practitioner. Doing just the physical exercises would give temporary benefits. This system cannot guide the user through all these stages because it requires years of experience, devotion and practice. Yoga is a complete science inherited from our ancestors and this work is only an effort to get people started.

6. Conclusion

This work is an implementation model of Agent Technology and an effort to impart intelligence in the way people see yoga, think about yoga and the

authenticity in doing it. The agent has been written taking authentic research works into reference so that the recommendations are in sync with traditional Indian yoga asanas. All efforts have been made to make sure that all traditional and cultural values associated with yoga are preserved. In further research works these agents may be used in different environments like android or IOS. They may even be deployed on a container placed on a central cloud network and work for distributed systems. Performance analysis has been done by taking feedback from users who have sufficient knowledge in this regard along with experience in doing yoga.

References

[1] H. R. J. Subalakshmi and N. Ch. S. N. Iyengar, "Enhancing a Traditional Health Care System of an Organization for Better Service with Agent Technology by Ensuring Confidentiality of Patients' Medical Information", Cybernetics and Information Technologies, vol. 13, no 3.
[2] C. J. Su and C. Y. Wu, "JADE Implemented Mobile Multi Agent Based, Distributed Information Platform for Pervasive Health Care Monitoring", Applied Soft Computing, vol. 11, **(2011)**, pp. 315-325.
[3] F. Bellifemine, A. Poggi and G. Rimassa, "Developing Multi-agent Systems with JADE", Proceedings of the 7th International Workshop on Intelligent Agents VII, Agent Theories Architectures and Languages, Springer-Verlag London, UK, **(2001)**, pp. 89-100.
[4] F. Bellifemine, A. Poggi, G. Rimassa and P. Turci, "An Object Oriented Framework to Realize Agent Systems", Proceedings of WOA 2000 workshop, Parma, Italy, **(2000)**, pp. 52-57.
[5] C. Woodyard, "Exploring the therapeutic effects of yoga and its ability to increase quality of life", International Journal of Yoga, vol. 4, no. 2, **(2011)**, pp. 49-54.
[6] A. Moreno, A. Valls, D. Isern and D. Sanchez, "Applying Agent Technology to Healthcare: The GruSMA Experience", Intelligent Agents in Healthcare, IEEE, **(2006)**, pp. 63-67.
[7] S. S. P. Gupta, "A Multi-Agent System (MAS) Based Scheme for Health Care and Medical Diagnosis System", proceedings of International Conference on Intelligent Agent and Multi Agent Systems, Chennai, **(2009)**, pp. 1-3.
[8] F. Bellifemine, L. G Caire and D. Greenwood, "Developing Multi-Agent Systems with JADE", John Wiley Sons, Ltd., England, **(2007)**.
[9] F. Bellifemine, A. Poggi and G. R Imassa, "JADE – A FIPA-Compliant Agent Framework", Proceedings of 4th Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, **(1999)**, pp. 97-108.
[10] S. Singh, T. Kyizom, K. P Singh, O.P Tandon and S.V Madhu, "Influence of Pranayamas and Yoga-Asanas on serum insulin, blood glucose and Lipid Profile in type 2 diabetes", Indian Journal of Clinical Biochemistry, vol. 23, no. 4, **(2008)**, pp. 365-36.
[11] B. M. Han, S. J. Song, K. M. Lee, K. S. Jang and D. R. Shin, "Multi-Agent System Based Efficient Healthcare Service", Proceedings of 8th International Conference on Advanced Communication Technology, vol. 1, **(2006)**, pp. 47-51.
[12] A. Woolery, H. Myers, B. Sternlieba and L. Zeltzer, "A Yoga Intervention for Young Adults with elevated Symptoms of depression", Alternative Therapies, vol. 10, no. 2, **(2004)**.

Authors

Abhishek Mathur, he is currently pursuing B.Tech in Computer Science and Engineering from VIT University, Vellore, Tamil Nadu, and India. His areas of interest are Artificial Intelligence, Game Development, Algorithm Designing, Database Systems and Networking.

G. Siva Shanmugam, he is working as assistant professor at School of Computer Engineering at VIT University, Vellore, TN, India. His research interest includes Distributed systems, Web Security, Sensor Networks, Mobile and Internet Computing. He has had 6 years of teaching experience and currently doing his PhD research on Green Cloud computing.

N. Ch. S. N. Iyengar, he is a senior professor at School of Computing Sciences and Engineering at VIT University, Vellore, TN, India. His research interests include Security in Networks, Information, Cloud, Agent computing, e-services and Fluid Dynamics (Porous Media). He had 30yrs of teaching and 20 yrs of research experience with a good number of publications in reputed International Journals & Conferences. He chaired many Intl. Conf. and delivered Key note lectures, served as PC member Reviewer. He is Editor in chief and also Editorial Board member for many Int'l Journals.