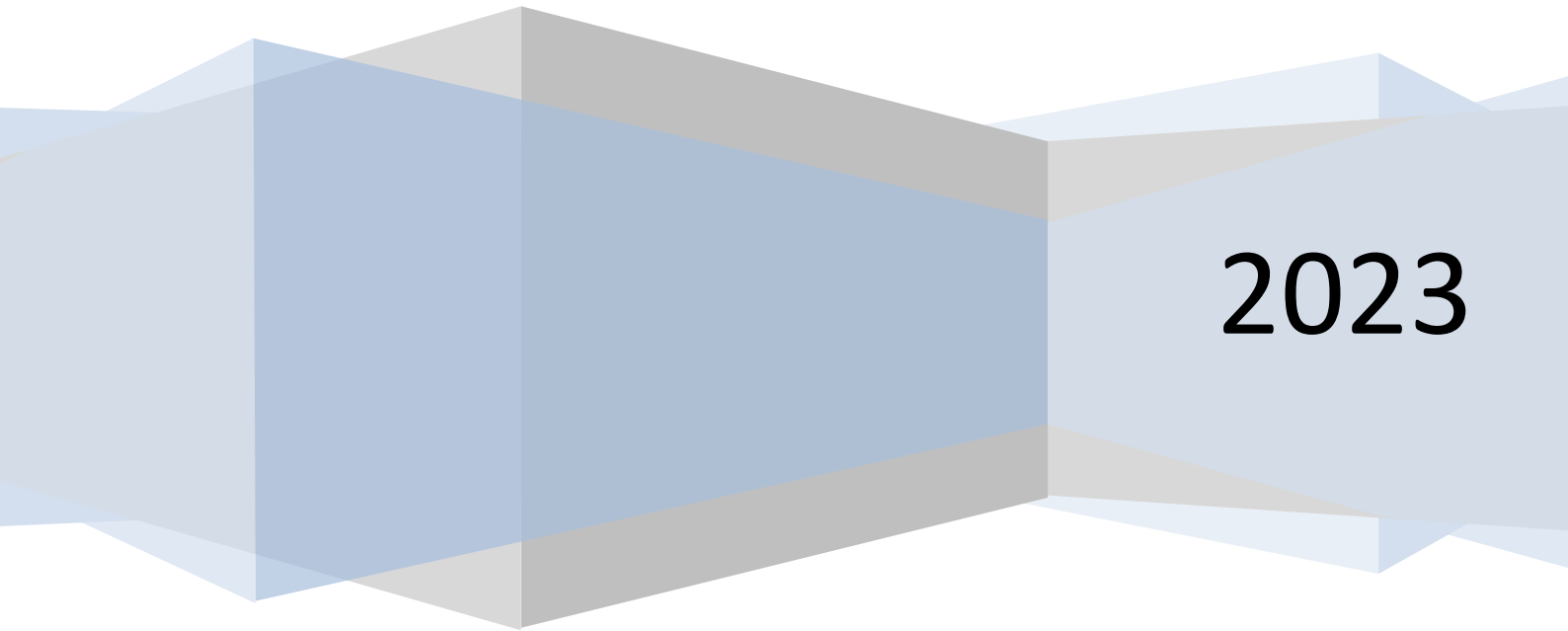
Intelligent Agent

Development Team Project

Design Proposal

Group 2



2023

1.0 Background

Computerised devices are being used in every part of our lives. With the exponential growth of digital data, businesses and individuals face the challenge of organising and categorising vast amounts of files. Intelligent agents can process and analyse large volumes of data much faster and more efficiently than humans (Russell, 2021). Also our digital assets are vulnerable to attack from malicious files. There are several file types that can be dangerous to our computer system (Hoffman, 2017) which may be difficult for an average computer user to differentiate between ordinary and malicious files. The aim of this project is to deploy intelligent agents to automate the process of classifying files. This project design proposal provides an outline plan for designing and developing intelligent agents to classify files based on the file name extension.

2.0 Objectives

The main objectives of this project are:

- To design and develop intelligent agents to be able to analyse the file name extension for classification for various file types.
- To develop an interface for user to interact with the agent to review and validate the classification results and provide feedback.
- To perform testing and validation to ensure accuracy and efficiency of the intelligent agents

3.0 Methodology

This project would deploy multi-agent system as it could be a powerful approach for file classification. Multi-agent systems involve the coordination and

interaction of multiple agents to achieve collective goals (Wooldridge, 2009). Peng et al. (2001) had compared single and multi-agent performance based on criteria such as response time, quality of classification and economic/privacy considerations for the classification of computer science and Medline documents. Their results indicate that collaborative multi-agent system performs better than single agent system.

Reactive type would be used for developing all the agents in this project as they respond to their environment based solely on the current input it receives. These agents are typically fast and efficient which directly percept the files and performs an action based on the current situation. (Russell, 2021).

3.1 Project Design

The project design using Unified Modelling Language (UML) has been presented to visualise the role of each agent in the multiple agent model. Figure 1 displays the Use case diagram UML presents a visual representation that show an overview of the actors participating in the system, the various functions required by these actors, and the interactions among these actors. Figure 2 displays the Sequence diagram UML, illustrating the interactions between objects and the sequential order in which these interactions take place.

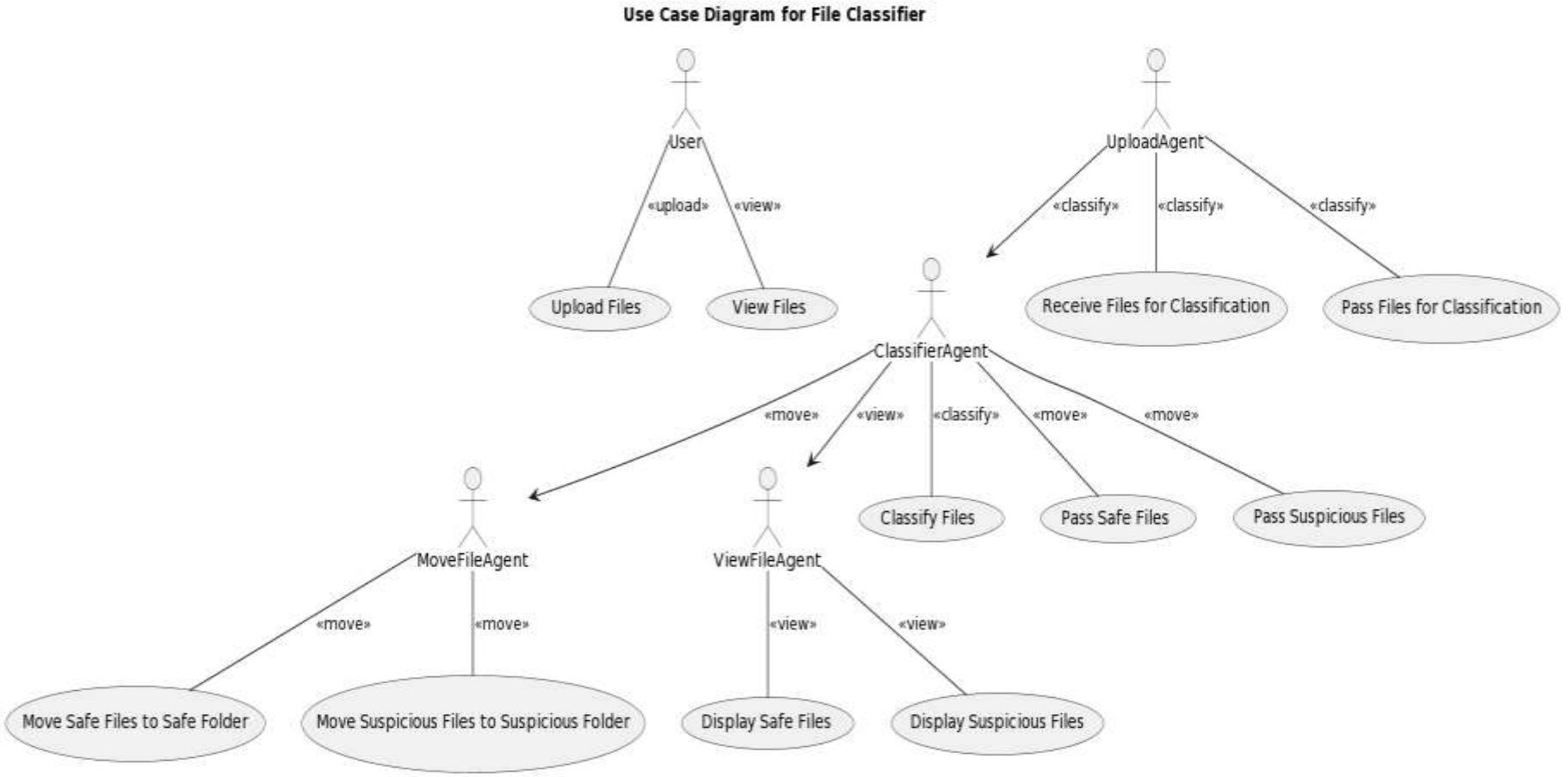


Figure 1: Use Case UML

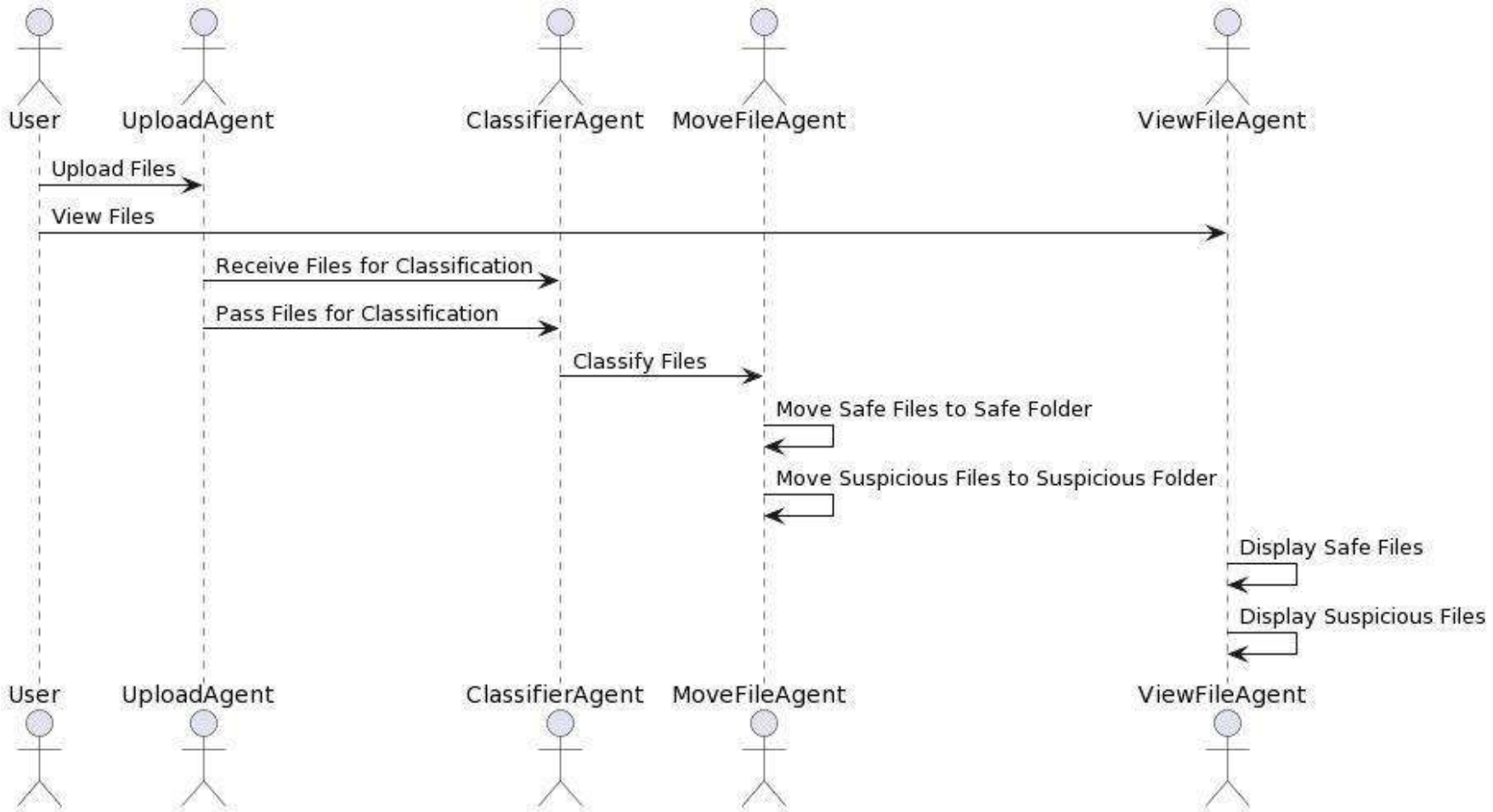


Figure 2: Sequence diagram UML

3.2 Development

User:

- uploads files: The User actor can upload files by interacting with the UploadAgent actor.
- views files: The User actor can view files by interacting with the ViewFileAgent actor.

The following four agents would be developed for file classification.

UploadAgent:

This agent would be responsible for accepting file uploads from the user through the web interface. It would use Flask's capabilities to handle file uploads securely.

- receives files for classification: The UploadAgent actor receives the files from the User and is responsible for receiving files for classification.
- passes files for classification: The UploadAgent actor passes the received files to the ClassifierAgent actor for classification.

ClassifierAgent

This agent's function is to classify the uploaded files based on their extensions. It would categorise files as either 'safe' or 'suspicious' based on a predetermined list of file extensions.

- classifies files: The ClassifierAgent actor is responsible for classifying the files it receives from the UploadAgent.
- passes safe files: The ClassifierAgent actor passes the classified safe files to the MoveFileAgent actor.
- passes suspicious files: The ClassifierAgent actor passes the classified

suspicious files to the MoveFileAgent actor.

MoveFileAgent

After classification, this agent would move files to their designated folders ('Safe' or 'Suspicious'), and within these, to specific sub-folders based on file type. It would use the Python 'os' and 'shutil' libraries for these operations.

- moves safe files to safe folder:
- moves suspicious files to suspicious folder:

ViewFileAgent

The ViewFileAgent actor is responsible for displaying the safe files and suspicious files to the User.

- displays safe files
- displays suspicious files

3.3 Testing and Evaluation

User interface would be created as shown in figure 3 and 4 which allows the user to input testing files, and the intelligent agent will provide classifications results as shown in figure 5.

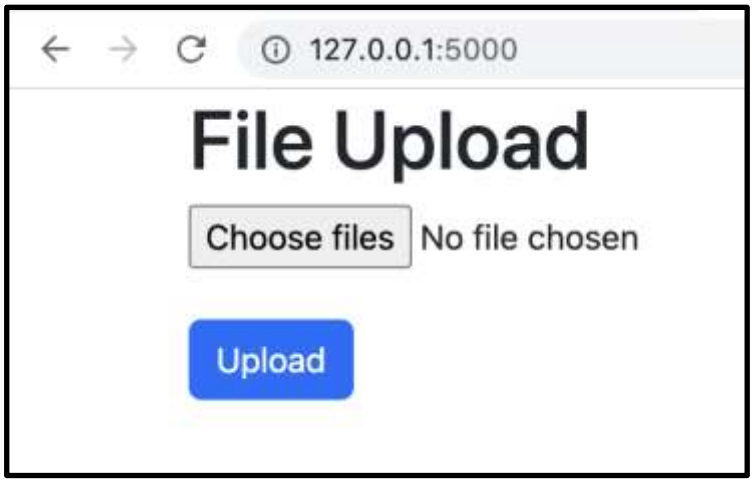


Figure 3: User interface for file upload

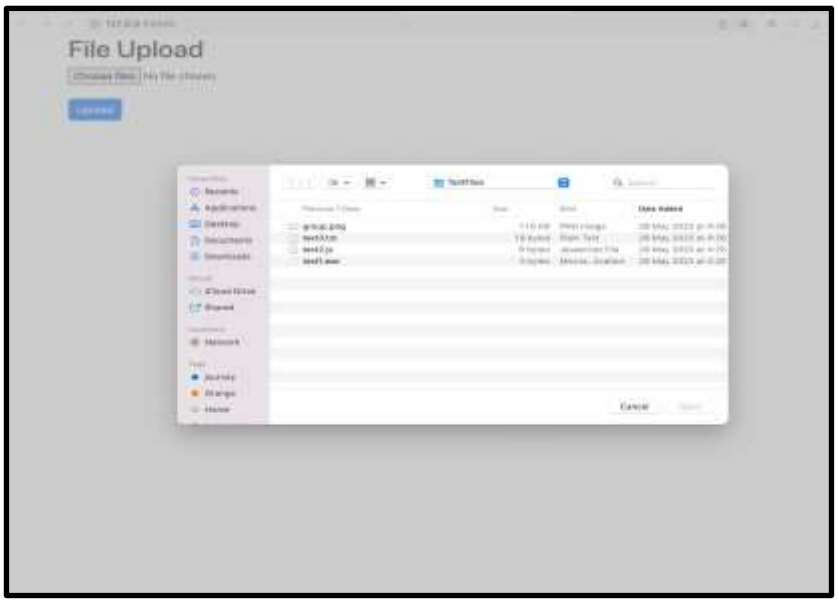
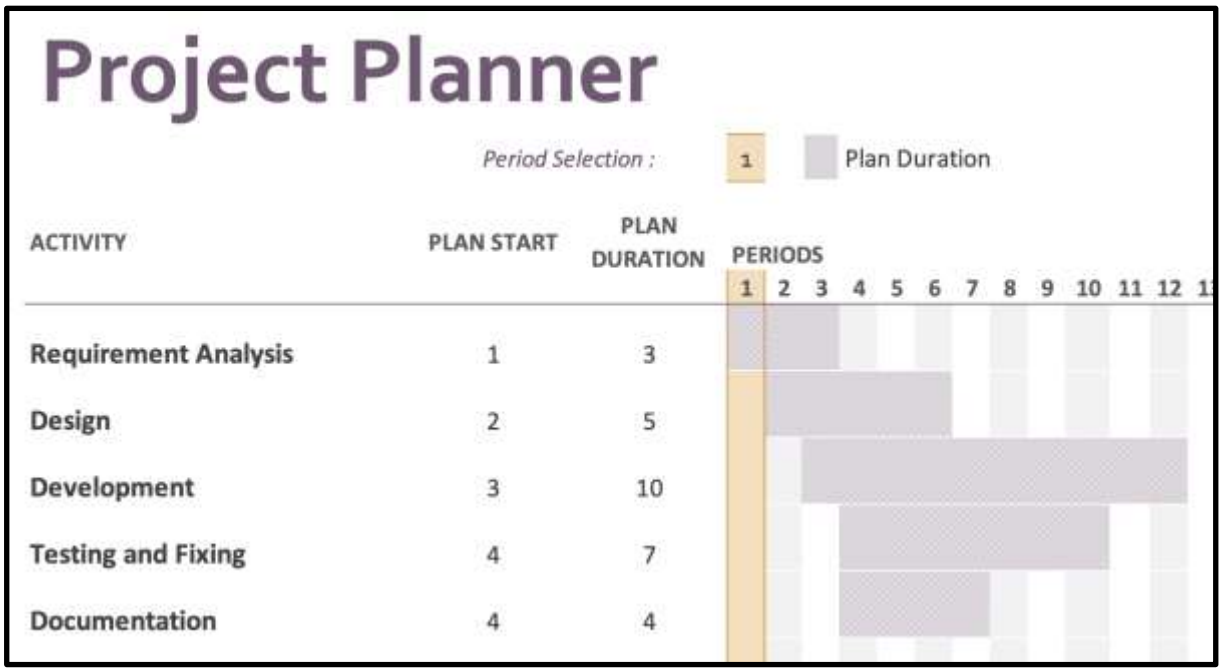


Figure 4: User interface displaying file explorer for file upload

Figure 5: Interface displaying classification results

4.0 Timeline



5.0 Resources

Project team	3-member student team
Training and testing data	Sample files such as documents, images, audio and video files (Examples: JPEG, PNG, PDF, DOCX, XLXS, MP3, WAV, etc)
Python libraries	**Flask** – ideal for building web interface for users to interact with the system, as it is light-weight and easy to use. It has libraries that can handle file uploads and can validate and store uploaded files.

	(Aggarwal, 2019). **Werkzeug** – useful for ensuring security in file handling and for handling communication between the server and the application, including tasks like template selection for requests and responses (Nokeri, 2022) **os** – useful for various operating system-dependent functionalities such as managing file and directory operations **shutil** – useful for performing high level operation on files and in automated file management

.

6.0 Challenges

A major challenge in the development process would be to warrant the accurate classification of files. The current approach of using file extensions can be deceptive if a malicious file is hidden with a benign extension (Casey, 2011). Further, ensuring secure file upload to prevent directory traversal attacks is a critical concern. Input validation and performing regular security testing could mitigate the risk of directory traversal attacks (Stuttard & Pinto, 2011).

7.0 Expected Outcomes

This project is expected to achieve the defined objective of developing well-designed intelligent agents capable of classifying files accurately based on their file name extension, providing improved data organisation and efficient file retrieval for users and reducing manual efforts in file management.

8.0 Conclusion

This project proposal outlines the plan for designing and developing an intelligent agent focused on file classification. By implementing this proposed project, the agents aim to automate the organisation and categorisation of files, thereby enhancing user experience in file management. Successful implementation of this project will contribute to streamlined file management and improved productivity in various domains.

9.0 References

Aggarwal, S. (2019) Flask Framework Cookbook. 2nd ed.. Packt Publishing. Available from: https://www.perlego.com/book/1031659/flask-framework-cookbook-over-80-proven-recipes-and-techniques-for-python-web-development-with-flask-2nd-edition-pdf [Accessed 10 June 2023].

Casey, E. (2011) Digital Evidence and Computer Crime – Forensic Science, Computers and the Internet, 3rd ed. Academic Press.

Hoffman., C (2017) 50+ File Extensions that are potentially dangerous on Windows. Available from: https://www.howtogeek.com/137270/50-file-extensions-that-are-potentially-dangerous-on-windows/ [Accessed 10 June 2023].

Nokeri, T.C. (2021) Web app development and real-time web analytics with Python: Develop and integrate machine... learning algorithms into web apps. S.l.: Apress.

Peng, S., Mukhopadhyay, S., Raje, R. & Palakal, M. (2001) A Comparison Between Single-agent and Multi-agent Classification of Documents. *Proceedings of the 10th Heterogeneous Computing Workshop* 2; 20090.2.

Russell, S., Norvig, P., Canny, J. & Malik, J. (2021) Artificial Intelligence: A Modern Approach 4th ed. Pearson.

Stuttard, D. & Pinto, M. (2011) The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws. 2nd ed. Wiley Publishing.

Wooldridge, M.J. (2009) An Introduction to Multiagent Systems. Chichester: John Wiley & Sons.