A Study on Game Design and Development using Unity and Unreal Engine: A Comparative Analysis of Features, Tools, and Performance

Omkar Mishra,
Department of Computer Science and Engineering,
Chandigarh University, Punjab, Mohali, India
Abhishek Raj,
Department of Computer Science and Engineering,
Chandigarh University, Punjab, Mohali, India
Jagatjeet Singh,
Department of Computer Science and Engineering,
Chandigarh University, Punjab, Mohali, India
Ansh Singh,
Department of Computer Science and Engineering,
Chandigarh University, Punjab, Mohali, India
Nidhika Chauhan,
Division of Research and Innovation, Uttaranchal University, Dehradun, India Email: nidhi29.chauhan@gmail.com5
Navneet Kaur,
Division of Research and Innovation, Uttaranchal University, Dehradun, India Email: navneetsehgal5@gmail.com
Satbir S Sehgal
Division of Research and Innovation, Uttaranchal University, Dehradun, India
Email: ssehgal@uumail.in7

Abstract—Game Development (GD) involves designing, developing, and producing video games. Concept development, story- boarding, programming, graphics, sound, and quality assurance testing are involved. GD's goal is to provide gamers a fascinating experience. Game creation requires imagination, technical competence, and project management. The work summarizes the GD process, emphasizing key factors and recommended practices for game development. GD begins with concept development and basic ideation, which we discuss in the article. We discussed storyboarding's role on game mechanics and storytelling. We also explore game programming technicalities, emphasizing the need to adopt relevant game engines like Unity or Unreal Engine to simplify development processes. Art design enhances GD's visuals and mood. We study character, environment, and animation design strategies. We also discuss how audio effects, background music, and voice-overs enhance immersion. Finally, GD quality assurance testing is crucial. Rigorous testing eliminates bugs, malfunctions, and playability difficulties, giving gamers a smooth, delightful experience. This study article illuminates game creation by synthesizing these crucial elements. It guides prospective game creators through critical considerations and best practices to produce compelling and successful games.
Index Terms—Game Engine, Game Development, Game designing.

I. INTRODUCTION

Create immersive and engaging gaming experiences by combining creativity, technological expertise, and problem- solving skills in the fascinating and quickly developing profession of Game development (GD). You can create worlds, give characters a life of their own, and influence how players engage with your game as a game producer [1].

Video game popularity and technological improvements have made game production a booming industry with limitless potential for innovation and expansion. There are many other career routes you might pursue in the field of GD, whether you're interested in designing mobile games, producing virtual reality experiences, or producing console games [2]. You must have a thorough understanding of programming languages, game engines, graphics, and sound design to become a game developer. However, it goes beyond technical. Johnson et al [1] pointed out that GD will gain worldwide usage.

II. RELATED WORKS

A great game always has the same thing in common: reaching out to the audience, having an articulate display of their ideas, and having deep logic for mechanics in the game. In October 1958, Physicist William Higginbotham created what is thought to be the first video game [3]. This is a complex and fascinating field that involves creating interactive experiences that engage players on a variety of levels. From the initial concept to the final product, GD requires a deep understanding of programming, design, art, music, and storytelling [4]. One of the most important aspects of GD is the gameplay itself. Good gameplay should be fun, challenging, and engaging, with a clear objective that the player can work towards [5]. To achieve this, game developers must carefully balance mechanics, difficulty, and pacing to create an experience that is both satisfying and enjoyable [4]. Another crucial element of GD is storytelling. Whether through cut scenes, dialogue, or gameplay, a game's narrative can greatly enhance the player's immersion and emotional investment in the experience. Strong storytelling can also help to create a sense of cohesion and purpose in the game world, and make the player feel like they are part of something larger than themselves. Of course, none of this would be possible without the technical expertise required to bring a game to life. Game developers must be skilled in programming languages like C++, Java, and Python, as well as familiar with game engines and development platforms like Unity, Unreal Engine, and Game maker Studio. They must also have an eye for design, with a deep understanding of color theory, typography, and layout [6].

In addition to technical skills, game developers must also be able to collaborate effectively with others. GD is a highly collaborative field, and success often hinges on the ability to work well with a team. This includes communicating effectively, being receptive to feedback, and being willing to compromise when necessary [4].

Overall, GD is an incredibly rewarding and challenging field that requires a broad range of skills and expertise. Whether you are interested in creating indie games or working on blockbuster titles, there is always something new to learn and explore in the world of GD [5].

III. GAME ENGINES AND RELATED TOOLS

Video games are a rising industry where people can connect with each other and react to a common environment. This typical environment can be created using a virtual environment. Video games have had the concept since their beginning of putting players in a critical situation. According to the pressman, the games are software programmed to de-liver entertainment [2]. Games There can be categories among three classes [3], which are console, mobile, and cross-platform. Each platform has their own software and UI, and based on that, their game play. One of the best GD engines are UNREAL and UNITY engines, these engines work on C++/blueprint and C-sharp respectively.

A. Unreal engine

It was developed by Epic Games around 25 years ago. Among the other main-stream game engines,

unreal engines stand out due to sheer number of tools it provides for all GD disciplines. Epic Games is the company that created and now maintains the game engine known as Unreal Engine. Due to its sophisticated features and tools that enable the design, development, and deployment of high-quality, immersive video games across various platforms, it has grown in popularity among game developers and creators. This is shown in Fig 1.

The sophisticated graphics rendering technology of Unreal Engine is one of its standout characteristics. It produces gorgeous graphics with realistic lighting and effects using a strong rendering pipeline. This enables game developers to design immersive settings that are interesting to play in.

The ability of Unreal Engine to simulate physics is another crucial feature. It is capable of simulating realistic physics, which includes how virtual world objects behave and interact with one another. This enables game designers to construct gameplay mechanisms that are more plausible and realistic, which enhances the overall gaming experience.

Additionally, Unreal Engine has sophisticated audio features such as support for spatial audio that contribute to the development of an immersive and realistic gaming environment. Additionally, it offers a selection of animation tools that enable the development of intricate character animations and movements [7].

One of the most unique features of Unreal Engine is its Blueprint visual scripting system. This system allows designers and developers to create complex game mechanics and logic without the need for traditional programming knowledge. It uses a drag-and-drop interface to create visual representations of game mechanics and behaviors, making it easier for developers to iterate on game design and experiment with different ideas.

Unreal Engine is a versatile and scalable tool, suitable for both indie developers and large game studios. It is used to create a wide range of games, including first-person shooters, role-playing games, racing games, and more. Additionally, Un-real Engine is also used in non-gaming applications such as architectural visualization, virtual training, and film and TV production.

Unreal Engine is available for free for non-commercial use, with a royalty-based licensing model for commercial use. This means that game developers can use Unreal Engine to create and distribute games without any upfront costs, but they will need to pay a percentage of their revenue to Epic Games if their game generates a certain amount of revenue.

Overall, Unreal Engine is a powerful and flexible tool that offers a wide range of features and tools to help game developers and creators create high-quality, immersive gaming experiences. It is constantly evolving with updates and new features, making it a popular choice among game developers and creators.

B. Unity Engine

Unity is a game engine developed and maintained by Unity Technologies. It has become a popular tool for game developers and creators due to its ease of use, cross-platform development capabilities, and advanced features and tools. This is shown in Fig2.

One of the key advantages of Unity is its accessibility. It has a user-friendly interface and a large community of developers who share resources and provide support to each other. This makes it a popular choice for indie developers and small studios who may not have access to the same resources as larger studios. Unity also offers extensive documentation and tutorials, making it easy for developers to get started.

Another important aspect of Unity is its cross-platform development capabilities. It supports the deployment of games to multiple platforms, including PC, con-soles, mobile devices, and VR/AR. This makes it easier for developers to reach a wider audience and maximize their game's potential revenue. Unity offers a range of features and tools that are similar to those found in Unreal Engine, such as advanced graphics rendering, physics simulation, audio, animation, and AI. It also provides a visual scripting system called Unity Scripting, which allows developers to create complex game mechanics and behaviors with-out the need for traditional programming knowledge.

Fig. 1. UnrealEngine

Fig. 2. Unity Engine

Unity's advanced graphics rendering system allows for the creation of stunning visuals with realistic lighting and effects. It supports high-quality graphics, including HDR, PBR, and real-time global illumination, which can help create immersive and engaging game environments.

Unity's physics simulation capabilities enable developers to create realistic physics, including the behavior of objects in a virtual world and the interactions between them. This allows for more realistic and believable gameplay mechanics that add to the overall game experience.

Unity also offers advanced audio capabilities, including support for spatial audio, which helps create a more immersive and realistic game environment. It provides a range of ani- mation tools that allow for the creation of complex character animations and movements.

Unity is not just limited to GD. It is also used in other industries, such as architecture, engineering, and construction. It is used to create virtual and augmented reality experiences, as well as simulations for training and education.

Unity is available for free for personal and small business use, with a range of paid plans available for larger businesses and organizations. Like Unreal Engine, it uses a royalty-based licensing model for commercial use.

Overall, Unity is a powerful and flexible tool that offers a range of features and tools to help game developers and creators create high-quality, immersive gaming experiences. Its focus on ease of use and cross-platform development makes it a popular choice for indie developers and small studios.

C. Blueprinting and visual scripting

Blueprint as illustrated in Fig3. is a visual scripting system in Unreal Engine that allows developers to create gameplay mechanics and other functionality without the need for traditional programming knowledge. With Blueprint, developers can create logic and behavior for game objects, characters, and environments using a drag-and-drop interface, similar to a flowchart. It is designed to be user-friendly and accessible to developers of all skill levels, from beginners to experts. It allows developers to create complex game mechanics without the need for complex programming code. Instead, developers can connect pre-built nodes and functions to create logic and behavior.

Blueprint also allows developers to quickly prototype game mechanics, test ide-as, and make changes in real-time. It integrates with other Unreal Engine tools such as the Material Editor and Level Editor, making it easy to create complex and detailed game environments.

Some of the key features of Blueprint include:

- Node-based visual scripting: Blueprint uses a node- based visual scripting system that allows developers to connect pre-built nodes and functions to create game mechanics and behavior.
- User-friendly interface: Blueprint is designed to be user-friendly and accessible to developers of all skill levels,

Fig. 3. Blueprint

with a drag-and-drop interface that makes it easy to create and modify game mechanics.

- Rapid prototyping: Blueprint allows developers to quickly prototype game mechanics and make changes in real- time, speeding up the GD process.
- Integration with other Unreal Engine tools: Blueprint integrates with other Unreal Engine tools such as the Material Editor and Level Editor, making it easy to create complex game environments.
- Debugging and error checking: Blueprint includes built-in debugging and error checking tools, making it easy to identify and fix issues in the game mechanics.

Overall, Blueprint is a powerful and versatile tool that makes it easy for developers to create complex gameplay mechanics and behavior without the need for traditional programming knowledge. It is a key component of Unreal Engine and a major reason why it is such a popular game engine among developers.

Unity does not have a native visual scripting system like Unreal Engine's Blue-print. However, there are several third- party visual scripting tools available for Unity that offer similar functionality. Here are a few examples:

- Bolt: Bolt is a visual scripting tool for Unity that allows developers to create gameplay mechanics and other functionality without the need for traditional programming knowledge. Bolt uses a node-based visual scripting system similar to Unreal Engine's Blueprint.
- PlayMaker: PlayMaker is a well-liked visual scripting tool for Unity that gives programmers a drag-and-drop inter- face for constructing intricate game systems. A visual scripting system based on state machines is used by PlayMaker.
- Behavior Designer is a visual scripting tool for Unity that is designed specifically for the creation of AI and behavior trees. Using a node-based visual scripting framework, it enables developers to construct intricate AI behaviors etc.

D. Video game graphic creation and history

- Gameplay, which excludes all audiovisual components, is the umbrella term used to define the whole experience a player has when participating in a video game. These encounters can range from engaging in conversation with an in-game character to slashing an opponent to death etc. [4].
- Video games for homes and arcades first became widely popular in the 1970s. In terms of graphic style, many games from the 1970s were fairly spare and abstract [4].
- As game developers started experimenting with new technologies and games became more complicated, the 1980s displayed a wide range of styles. There were several successful abstract games at this time. One of these games was the previously mentioned Tetris, which continues to be the best- selling game in the entire world [4].
- For video games, the 1990s were an inventive decade. The decade saw a rise in the use of handheld gaming devices and home video game con-soles. The change from sprite to 3D game visuals was another major development during this time [4].

E. UI design tools

The UI components as shown in Fig 4. that the tools target include: The majority of useful tools and technologies concentrate on a specific area of the user interface (UI) that has a big problem and might be solved comprehensively and efficiently. UIMSs, model-based and automatic generation techniques, and other strategies are examples of methodologies that have failed to achieve commercial success as a result of problems associated with attempting to address the entire problem [5].

IV. METHODOLOGY OF GD

A. Game Concept

We notice parallels with board games when developing, which involve the inter-action of numerous actors who are all interested in changing an artefact. In the games we've described, several players interact with a configuration on a board. Players must fit their pieces into an existing field, just like when they are designing. Rules, conventions, and principles restrict how they can move, and they must come to flexible, negotiating agreements on which conventions and rules to apply in a particular circumstance. Last but not least, players predict the configurations that will be built. Our games are

- analogous to actual design situations in this way. However, unlike real-life experience, the games give us a manipulable and well-defined context in which to analyses design actions [6].

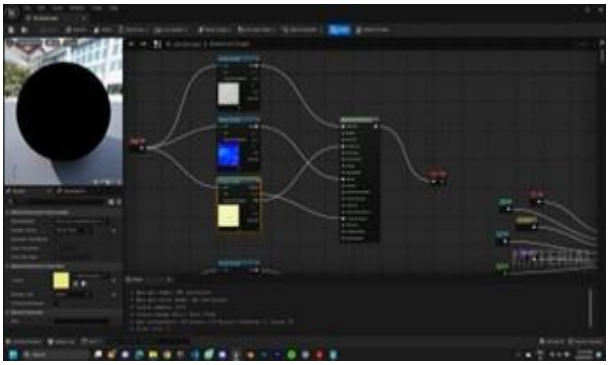

Fig. 4. Unreal UI

B. Designing a Game

Unreal Engine as it requires a team of developers and designers with expertise in various areas such as game design, programming, art, animation, and sound de-sign. However, the paper provides a general overview of the process involved in making a game using Unreal Engine.

- Install Unreal Engine: The first step is to download and install Unreal Engine on your computer. You can do this by visiting the Unreal Engine website and clicking on the "Get Started" button. Follow the instructions to download and install Unreal Engine.
- Create a New Project: Once you have installed Unreal Engine, open it and create a new project. You can choose a template or start from scratch. The template will have pre-built assets and settings that you can use as a starting point for your game.
- Design the Game: The next step is to design the game, which involves creating the game world, characters, and gameplay mechanics. You can use Unreal Engine's Blueprint visual scripting system to create game mechanics without needing to know how to code.
- Import Assets: Once you have designed the game, you can import assets such as 3D models, textures, and sound effects into Unreal Engine. You can create these assets yourself or purchase them from third-party marketplaces.

Set up Levels: The next step is to set up levels. Levels are different sections of the game world that the player will progress through. You can create levels using Unreal Engine's level editor, which allows you to add assets, terrain, lighting, and effects.

Run the level: The player should check the working of logic in the game for a smoother experience of the game.

Creating a game using Unreal Engine is a complex process that requires expertise in game design, programming, and art. However, Unreal Engine provides a range of tools and features that make it easier to create high-quality games with stunning visuals and immersive gameplay mechanics.

Programming

Computer programmers can treat other program as their data using a programming method called metaprogramming. It implies that a program may be created with the ability to read, produce, analyses, or convert another program, as well as to modify itself while it executes.

To create games for the meta-framework a set of metaprogramming scripts able to the GSL (Game Specification Language). For Metaprogramming script, these are described by Extensible Stylesheet Language (XSL) files, able to translate GSL specifications [10].

C. Prototyping a game

Before moving on to full development, a game's gameplay mechanics and features are tested and refined through the process of game prototyping. In order to make the necessary adjustments to the game's concept and detect any potential problems or flaws, prototyping is a crucial step in the GD process [9].

Here are some key benefits of game prototyping:

- Testing gameplay mechanics: Prototyping allows developers to test the basic gameplay mechanics of the game, such as character movement, combat, and puzzle-solving. This helps to identify any issues with the game mechanics early on and make adjustments as needed.
- Refining game features: Prototyping also allows

developers to experiment with different game features and refine them based on player feed-back. This can include features such as level design, AI behavior, and user interface design [10].

Early detection of potential issues: Before full development is undertaken, prototyping aids in the early detection of potential issues or design defects. By avoiding expensive and time-consuming changes later in the development process, this can ultimately save time and resources.

Cost-effective: Prototyping is typically less expensive than full development as it involves
- creating a simplified version of the game. This can help to save resources and reduce the risk of investing in a game that may not be successful.

Overall, game prototyping is an important part of GD that allows developers to test and refine the game's design before
- moving on to full development. By identifying potential issues early on, developers can save time, resources, and improve the overall quality
- of the game.

V. PROBLEMS AND SOLUTION

- Difficulty issues: In light of the material we read, the fol- lowing hypotheses were put forth to aid in our quest to identify patterns among various player types and the underlying causes of such patterns [5]. Skilled athletes will evaluate
- themselves more accurately it makes sense that seasoned players would have the better self-evaluation since they are more likely to be talented and trained. As a result, they ought to be more adept at metacognition than casual gamers. Since some casual players are likely to be skilled while others are not, the casual group's worst performers should have the worst perceptions of their own abilities. No matter how skilled they are, casual players will select easier challenges Regardless of their real skill level, players are inclined to choose relatively easy challenges because the casual player profile frequently suggests ulterior objectives. However, seasoned players will they prefer the challenge, thus they choose harder or more moderate problems. Novice players will choose lower difficulty levels, while expert players won't when the game's difficulty increased, Gilleade, Dix, and Allanson [9] showed that novice players got frustrated much more quickly than experienced players. As a result, casual gamers who opt for easier
- challenges will enjoy the game considerably more than those who go for harder challenges. However, because they like a challenge, experienced players will not enjoy easier difficulties if they don't match their skill level. This can be helped with giving a level Selection in menu [11-14]. While changing all the logics for different difficulty level or
- increase/decrease the pace of the game (include the damage given or Taken etc.).

Logic issues: These are programming errors that result in the game not functioning as intended, such as bugs that cause characters to move in the wrong direction or objects to disappear unexpectedly. This issue is solved by changing the logic time-to-time during prototyping.

Programming issues: This includes slow frame rates, long load times, and lagging, which can impact the overall game- play experience.

Memory leaks: These occur when a program doesn't release memory it has allocated, which can cause the game to slow down or even crash.

Compatibility Issues: These can arise when developing games for multiple platforms, such as mobile devices and PC, as different platforms may have different hardware configurations and operating systems.

Integration Issues: These occur when trying to integrate different components of the game, such as graphics, audio, and physics engines, which can lead to glitches and inconsistencies.

VI. MARKETING

As we discussed earlier, video games are a source of entertainment for the target audience. In the end, the product can be marketed off to online seller platforms like Steam, Epic Games, etc. A game without a target audience is a game without any fun. So, during game-development phase developers are concerned about the game's image and the way it feels.

A. Market

The production, promotion, and distribution of

video games across numerous platforms, such as consoles, PCs, and mobile devices, make up the multi-billion-dollar game sector. With so many businesses striving for market share, the market is fiercely competitive. Companies like Sony, Microsoft, Nintendo, and Valve, which create and sell gaming consoles and other hardware, are some of the biggest players in the market for video games. Other major contenders include Electronic Arts, Activision Blizzard, and Ubisoft, which develop and publish the actual games. The process of creating a video game involves many steps, including conception, design, prototyping, programming, testing, and deployment. Overall, the game market is a dynamic and ever-evolving industry that offers a wide range of opportunities for developers, publishers, and players alike [15-19].

B. GD and Gaming Communities

The free and open-source software (FOSS) methodology enables groups of like-minded people to create software systems and related artefacts that are supplied as open-source freeware rather than closed-source for-profit goods [4]. Soft- ware maintenance—adding and subtracting system functional- ity, debugging, restructuring, tuning, conversion (for example, internationalization), and migration across platforms—is a widespread, recurring process in FOSS development communities. The games are depicted in Fig 5 and Fig 6.

VII. CONCLUSION

The field of GD is one that is expanding quickly and has become very well-known. The history, technology, and design of game production have all been covered in this research study. Our research has shown that game production is a challenging process that calls for a blend of technical know-how, artistic ability, and user-centered design approaches. Through our research, we have determined several important elements that affect a game's success, such as interesting gameplay, compelling narrative, and cutting-edge game mechanics. We have also highlighted the opportunities and challenges that developing technologies bring for GD in the future.

Our research has shown that the demands and expectations of players drive the dynamic and ever-evolving industry of game creation. In order to produce games that are pertinent to and interesting to their target audience, it is crucial for game creators to keep up with the most recent trends and advances in the industry. Our research has shown that the demands and expectations of players drive the dynamic and ever-evolving industry of game creation. In order to produce games that are pertinent to and interesting to their target audience, it is crucial for game creators to keep up with the most recent trends and

Fig. 5. A basic car racing template: this image identifies the basic initial work for a GD

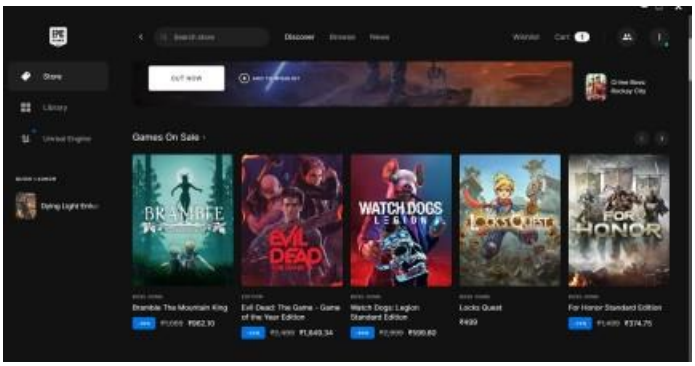

Fig. 6. Epic Games

advances in the industry. Overall, our research has highlighted the importance of GD as a creative and innovative industry that has the potential to shape the future of entertainment and education. We hope that our findings will contribute to the ongoing discussion and research in the field, and inspire others to explore the exciting world of GD.

REFERENCES

[1] Sharma, N.; Mangla, M.; Yadav, S.; Goyal, N.; Singh, A.; Verma, S.; Saber, T. A sequential ensemble model for photovoltaic power forecasting. Comput. Electr. Eng. 2021, 96, 107484.

[2] V. Singhal et al., "Artificial Intelligence Enabled Road Vehicle-Train Collision Risk Assessment Framework for Unmanned Railway Level Crossings," in IEEE Access, vol. 8, pp. 113790-113806, 2020, doi: 10.1109/ACCESS.2020.3002416.

[3] Dash, Sonali, Sahil Verma, Kavita, Md. Sameeruddin Khan, Marcin Wozniak, Jana Shafi, and Muhammad Fazal Ijaz. 2021. "A Hybrid Method to Enhance Thick and Thin Vessels for Blood Vessel Segmentation" Diagnostics 11, no. 11: 2017. https://doi.org/10.3390/diagnostics11112017

[4] G. Ghosh, "Kavita, Sahil Verma, NZ Jhanjhi, "Secure surveillance system using chaotic image encryption technique" 2020, Vol. 993, 012062," in IOP Conference Series: Materials Science and Engineering, vol. 993, no. 1, p. 012062.S. H. Kok, A. Abdullah, and N. Z. Jhanjhi, "Early detection of crypto- ransomware using pre-encryption detection algorithm," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 5, pp. 1984–1999, 2022.

[5] Shafiq, M., Ashraf, H., Ullah, A., Masud, M., Azeem, M., Jhanjhi, N., & Humayun, M. (2021). Robust cluster-based routing protocol for IoT-assisted smart devices in WSN. Computers, Materials & Continua, 67(3), 3505-3521.

[6] Lim, M., Abdullah, A., & Jhanjhi, N. Z. (2021). Performance optimization of criminal network hidden link prediction model with deep reinforcement learning. Journal of King Saud University-Computer and Information Sciences, 33(10), 1202-1210.

[7] Adeyemo, V. E., Abdullah, A., JhanJhi, N. Z., Supramaniam, M., & Balogun, A. O. (2019). Ensemble and deep-learning methods for two-class and multi-attack anomaly intrusion detection: an empirical study. International Journal of Advanced Computer Science and Applications, 10(9).

[8] Humayun, M., Jhanjhi, N. Z., Alruwaili, M., Amalathas, S. S., Balasubramanian, V., & Selvaraj, B. (2020). Privacy protection and energy optimization for 5G-aided industrial Internet of Things. IEEE Access, 8, 183665-183677.

[9] Khalil, M. I., Jhanjhi, N. Z., Humayun, M., Sivanesan, S., Masud, M., & Hossain, M. S. (2021). Hybrid smart grid with sustainable energy efficient resources for smart cities. sustainable energy technologies and assessments, 46, 101211..

[10] Hamid, B., Jhanjhi, N. Z., Humayun, M., Khan, A., & Alsayat, A. (2019, December). Cyber security issues and challenges for smart cities: A survey. In 2019 13th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS) (pp. 1-7). IEEE.

[11] R. Khan, J. Teo, M. A. Jan, S. Verma, R. Alturki and A. Ghani, "A Trustworthy, Reliable, and Lightweight Privacy and Data Integrity Approach for the Internet of Things," in IEEE Transactions on Industrial Informatics, vol. 19, no. 1, pp. 511-518, Jan. 2023, doi: 10.1109/TII.2022.3179728.

[12] Ramisetty, S.; Anand, D.; Verma, S.; Alaboudi, A.A. SC-MCHMP: Score-Based Cluster Level Hybrid Multi-Channel MAC Protocol for Wireless Sensor Network. In Information Security Handbook; CRC Press: Boca Raton, FL, USA, 2022; pp. 1–18.

[13] Kaur, N.; Devendran; Verma, S.; Kavita; Jhanjhi, N. De-Noising Diseased Plant Leaf Image. In Proceedings of the 2022 2nd International Conference on Computing and Information Technology (ICCIT), Tabuk, Saudi Arabia, 25–27 January 2022; pp. 130–137.

[14] Rani G, Oza MG, Dhaka VS, Pradhan N, Verma S, Rodrigues JJ (2020) Applying deep learning for genome detection of coronavirus. Res Sq. https://doi.org/10.21203/rs.3.rs-93564/v1

[15] Hazem Hanbal , Saad Metawa, Study on reasons of Failure of Small and Medium Enterprises: Looking into Egypt case, American Journal of Business and Operations Research, Vol. 0 , No. 1 , (2019) : 43-54

[16] Ngoc Minh Chau, Nguyen Thi Lan, Nguyen Xuan Thao, A New Similarity Measure of Picture Fuzzy Sets And Application in pattern recognition,

American Journal of Business and Operations Research, Vol. 1 , No. 1 , (2020) : 5-18

[17] K. Shankar, Fuzzy Clustering and Classification based Iris Recognition: A Medical Application, American Journal of Business and Operations Research, Vol. 1 , No. 1 , (2020) : 19-27

[18] Shilpi Pal , Avishek Chakraborty, Triangular Neutrosophic-based EOQ model for non-Instantaneous Deteriorating Item under Shortages, American Journal of Business and Operations Research, Vol. 1 , No. 1 , (2020) : 28-35

[19] Maha Saad Metawea, The role of financial institutions in supporting entrepreneurial success: Case of Egypt, American Journal of Business and Operations Research, Vol. 1 , No. 1 , (2020) : 36-51