Woodbridge VA 22193
+1-(571)-513-9489

# S. TALHA SHAHAB

Tshahab@gmu.edu
syedtalhashahab.github.io

## EDUCATION

Gar-Field High School | Woodbridge, VA *Aug 2018 - 22*

- High School Diploma. **GPA 4.2**

George Mason University | Fairfax, VA *Aug 2022 - 26*

- B.A in Applied Computer Science. Coursework includes Discrete, Data structures and algorithms, Calculus 2, Statistics
- **GPA 2.7** | Tennis, Basketball, and Swim. |
- Seeking research credit, internships for corporations, startups, or small businesses

## SKILLS

- Languages: | Java | Python | HTML | CSS | JavaScript | Processing | Kotlin |
- Tools & Frameworks: Django.py | Junit Testing | GIT | Node.js Runtime| React.js Library| Java Spring Boot |

## PROJECTS AND TECHNICAL

- **Wordle - JavaScript game** Jan /2023
  - Architected transitions and feedback to handle game states, selection, and validation in word list
  - Designed a custom virtual keyboard for user interactions and used data filtration to create tiles and keyboard layout
- **Endless Sand Type - Python game** Feb /2023
  - Added Front end visuals using library imports and back-end calculations using trigonometry for collisions
  - Optimized I/O operations containing 20,000 words and used Parallax scrolling effect for depth and user experience
- **Riu Plaza Stays - Java app** Mar /2023
  - Gathered client consultations to define project scope and client approval at each project phase
  - Established and documented specific performance criteria and used modern practices for a high-quality solution
- **Candy Crush - Python game** Apr /23
  - Designed using a unique algorithm for candy matches, slide detection, and event handling for users so for smooth gameplay
  - Utilized libraries and techniques for performance, using timers, and real-time rendering
- **Sudoku - Kotlin game** Jun /2023
  - Engineered using JavaFX with intertwined functionalities, managed threads, UI updates, for a smooth and responsive interface
  - Enhanced user experience using dynamic UI elements like non and editable cells and checking
  - Used Problem-Solving Skills and optimizing code for better performance and maintainability
- **Chrome Dino - Processing game** Sep /2023
  - Used object-oriented principles for real-time interaction via frame rate optimization and collision detection
  - Developed a comprehensive 2D game with intuitive fluid user controls and gameplay mechanics
  - Randomized spawning obstacles and game unpredictability for game entities dinosaur, birds, and cacti
  - Created custom gravity and simulations for jumping and movement for the player character
- **Space shooter - Python game** Oct /2024
  - Engineered game interface with custom-designed graphics for a scoring system with dynamic updates
  - Rendered loops and event handling and used collision detection algorithms for game entity interactions
  - Used object-oriented for technical sophistication, modular for game structure, and dynamic alien behavior for levels
- **Blackjack - JavaScript game** Mar /2024
  - Used object lookups for dynamic gameplay and translated rules int clean and reusable code
  - Tracked project from concept to completion using modular designs for easy maintenance, code readability and reusability
  - Developed card value calculations, ace handling, and win/loss via data structure for DOM manipulation and event handling
  - Created robust shuffling algorithm leveraging random index swapping for unpredictability and via user inputs (hit and stay)
- **Chess - Python game** May /2024
  - Created dynamic and interactive GUI with enhanced user experience and responsiveness and seamless play
  - Utilized modular events to handle various piece movements (pawn, rook, knight, bishop, queen, king)
  - Applied version control for open-source contributions and validation to detect check conditions for easy differentiation
  - Incorporated scaled images and strategic use of colors for depth and comprehensive view of the game state
  - Ensured cross-platform compatibility using event-driven programming model
- **Bit Defender Lite - Python encryption** Jun /2024
  - Developed a secure authentication using Tkinter, Fernet for data secrets, and used cryptographic key management
  - Prevented duplicates using data integrity in names and passwords and used encapsulation and abstraction
  - Used file handling to store credentials plus prevent duplicate usernames and managed error handling into modules for scalability
- **Mine Sweeper - Java game** Jul /2024
  - Randomized mine placement across the board and used Swing for tile and mine placement, manipulation, and recursion
  - Utilized conditional checks and boundary validations to handle edge cases such as tile clicks outside the board dimensions
  - Implemented event-driven programming using interfaces, flagging functionalities and encapsulated class properties and behaviors
- **2048 - Python game** Nov /2024
  - Used abstraction, polymorphism, inheritance, and dependency inversion principle
  - Employed interface segregation principal and used method overriding, and private variables