# Jawad Arshad

647-782-1120 | arshad.jawad1942@gmail.com | jawadarshad.ca | github.com/HamJaw1432

## Education

**University of Toronto (GPA: 3.7)** Toronto, ON

*Bachelor of Science in Computer Science, Specialist Program Comprehensive Stream* *Sept. 2019 – May 2024*

## Experience

**Full Stack / Blockchain Developer** May 2022 – Present

*Dapp Technology Inc* *Toronto, ON*

- Built a full-stack web application using **NextJS with Typescript**, **NestJS** and **Firebase** to allow users to bridge tokens on multiple chains, had over **30+ chains with 4000+ tokens on 15+ bridges.**
- Developed a DeFi Hub application that allowed users to trade, earn and explore in DeFi, with **real time price data shown on a Tradingview lightweight charts.**
- Implemented functions to **firebase function** that completed various tasks, like updating a price document on **firebase store** on a scheduled basis.
- **Deployed** many Smart Contracts to the **blockchain**, like a presale contract, and a lottery contract.

**Machine Learning Teaching Assistant** Jan 2024 – Apr 2024

*University of Toronto* *Toronto, ON*

- Taught two Classes per week various topics in machine learning like **regression, classification, clustering, dimension reduction, Bayesian methods, etc.**
- **Worked with the teaching team** to provide a holistic learning experience
- Contributed to the **assessment and evaluation process** through grading assignments, invigilating exams, as well as **providing feedback**.
- **Provided extra help to students** by holding office hours for them to ask questions.

**Quality Assurance Co-op** May 2021 – Jan 2022

*Clearbridge Mobile* *Toronto, ON*

- Developed scripts in **Java** with **selenium** to automate web test flows to significantly reduce the time it takes to start testing.
- Worked in teams of 6 – 15 member teams in an **agile** work environment to **develop, test, and release new features every couple of weeks.**
- **Tested** applications using a variety of test methods like, **regression, smoke, sanity and more**.
- Ensured that developed applications were working as expected for a large **enterprise company** client.

**Infantry Solider Primary Reserve** Oct 2018 – Aug 2021

*Canadian Armed Forces* *Toronto, ON*

- Faced challenges and **made critical decisions under pressure**, showcasing sound judgment and problem-solving skills
- **Communicated** and received orders via the Chain of Command.
- Always maintained **strict discipline** and adherence to military protocols and regulations.
- **Worked in teams** from 10 to 40 soldiers to accomplish various mock missions from the Chain of Command.
- **Led section/course** as the section/platoon senior to **effectively delegate/complete tasks** that were assigned

**Operations Associate** Aug 2020 – May 2021

*Staples* *Toronto, ON*

- **Improved efficiency** by improving time it took to complete daily tasks to allow for more time to do tasks that improved the store.
- **Oversaw inventory control**, including procurement, and stock tracking. Maintained accurate records and minimized stockouts, ensuring consistent availability of essential supplies.
- **Provided excellent customer service** by addressing inquiries and concerns related to product availability, orders, and inventory.

**Back-of-House Staff** Oct 2019 - Jan 2020

*Pizza Hut* *Toronto, ON*

- **Improved efficiency** of the back of the store by 50% by improving the way we schedule when to complete tasks in what order.
- Prepared pizza dough, sauces, and toppings according to standardized recipes and **quality standards, ensuring consistency** in taste and presentation.
- Ensured the timely and thorough cleaning of dishes, utensils, and kitchen equipment, maintaining a smooth workflow in the restaurant's kitchen.

## Projects

**Pac-Man and Other AIs** | *Python* Jan 2024 – Apr 2024

- **Created a artificial intelligent pac-man agent** that explored paths through the mazes, to both reach a particular location and to collect all food efficiently. Used **DFS, BFS, UCS, varying the cost function, A*** (with many admissible heuristics, and even good heuristics for sub optimal performance, but better speed).
- **Engineered a artificial intelligent pac-man agent**, that completed the pacman game while avoiding the adversarial ghost. Agents included **reflex agents, Minimax** (with and without alpha-beta pruning), **Expectimax** (with different evaluation functions).
- **Built artificial intelligent agents to use Markov Decision Processes and reinforcement learning** to solve various problems, like grid world, simulated robot controller (Crawler) and Pacman.
- **Developed artificial intelligent agents** to hunt ghosts down in ghost busters, used **bayes nets and factors to infer the location of the ghosts, with variable elimination**.

**CTF Challenges** | *Python, C, Bash, Wireshark* Jan 2024 – Apr 2024

- **Hacked networks** to conduct different types of attacks, like **packet sniffing** (with wireshark, and scapy python libaray), **malicious gateway, APR spoofing, ICMP Ping flood, TCP-syc flooding, TCP connenction reset, DNS cache poisoning, http sniffer, ssl striping, as well as doing general penetration testing** (using nmap, openvas, metasploit).
- Looked at rainbow tables to reverse hashed passwords.
- **Hacked systems** using different attacks like **buffer overflow** (Stack Smashing Attack (Branching), Stack Smashing Attack (Shellcode), Adjacent Memory Shellcode), **TOCTOU**, **hidden malware** (malware inside a Debian package).
- **Hacked web applications** using various attacks like **Incomplete Mediation**, **SQL injection** (SQLi), **Cross-Site Scripting** (XSS), **Cross-Site Request Forgery** (CSRF).
- Wrote a protocol that is similar to TLS version 1.3, that used the **Diffie Hellman key exchange algorithm**.
- Worked with different public and symmetric key algorithms like **Caesar Ciphers, Substitution Ciphers, Polyalphabetic Ciphers, One Time Pad Ciphers, Stream Ciphers** (RC4, Salsa20/Chacha20), **Block Ciphers** (DES, AES), **RSA, ECC**. Also **exploited** different vulnerability like key reuse attacks, known plain text, etc.
- Explored cryptographic hash functions and message authentication codes (MD5 SHA-1, etc), and exploited different vulnerability like hash length extensions. Also used GPG to encrypt and sign messages.

**Langchain** | *Python, Git* Sept 2023 – Dec 2023

- **Contributed** to an **open-source python** repository called Langchain.
- Developed new tools that allowed **AI agents** to access information on the **google such as google jobs, google finance, google trends, and google lens** by calling a **external API** and parsing the data for the AI agent.
- **Wrote test cases and example docs** for the new tools.

**Collaborative Web Builder App** | *React, Express, MongoDB, Docker, Web Sockets, Git* Sept 2023 – Dec 2023

- Built a drag and drop website builder with **React and expressJS**, that allowed multiple users to work on the same site and chat using **web sockets** and allowed them to share their videos and audio with **webRTC**.
- Used **Cookies** to store user sessions, and **OAuth** to authenticate users.
- **Deployed** the website on a **google cloud virtual machine** using **docker images** for the frontend and backend.
- Deployed using a **nginx reverse proxy** to have both the front and backend on the same machine.
- Wrote a shell script to help **automate the deployment** process.

**Mock AirBnb Database and Backend** | *Java, MySql* May 2023 – Aug 2023

- Developed a replica database of Airbnb in **MySQL**.
- Allowed users to add listings with multiple amenities for any dates and allowing other users to rent out the listings
- **Normalized** the relation into **3NF**

- Built a **backend Java application** that allowed users to **CRUD** items into the database.

**Mock Twitter Database** | *MySql* May 2023 – Aug 2023

- Designed a mock Twitter **mySQL** database allowing users to create tweets and comments, like other users' tweets and comments, and retweet others' tweets.

**Hype Web App** | *React, MongoDB, ExpressJS, NodeJS, Jira* May 2023 – Aug 2023

- **Collaborated with a startup founder** to build a **MERN** platform facilitating interaction between students and companies for internships and competitions.
- Developed a **short polling chat application** within the platform for users/companies interaction.
- **Automated the deployment process using Git actions** to build **Docker images** and push them to Docker Hub.
- Used **Git Flow** including merging, rebasing, and PRs, and release management.
- Tracked team progress with **2 weeks agile sprints**, and **Jira** as a management tool
- Scoped project by creating user stories, use cases, personas, acceptance criteria, and estimating stories

**Machine Learning News Article Classifier** | *Python* Jan 2023 – Apr 2023

- Implemented and evaluated classifiers for labeling news articles into five categories using **Naive Bayes, Gaussian class conditional, and k-nearest neighbors classifiers**.

**Machine Learning Audio Emotion Classifier** | *Python* Jan 2023 – Apr 2023

- Developed and evaluated an **SVM classifier** and an **SVM classifier with PCA** to predict whether the emotion associated with a human audio signal is positive or negative using Mel spectrogram and chromagram features.

**Machine Learning Image Imprinter** | *Python* Jan 2023 – Apr 2023

- Applied **radial basis functions** to restore corrupted images in a painter program.

**Machine Learning Orange Classifier** | *Python* Jan 2023 – Apr 2023

- Wrote a basic **logistic regression** script to classify objects as oranges based on width and height.

**Machine Learning University Admission Predictor** | *Python* Jan 2023 – Apr 2023

- Developed and compared models for university admission prediction using **linear regression and k-means++**, including clustering the data before applying linear regression.

**Pintos Operating System** | *C* Jan 2022 – Apr 2022

- Enhanced the basic Pintos OS by incorporating features like **context switching between threads, support for user programs, virtual memory management, and implementation of a file system**.
- Improved Pintos to efficiently handle multithreading by managing interrupts, implementing context switching with a **Multilevel Feedback Queue scheduler**, and ensuring proper synchronization using **locks and semaphores** to avoid race conditions.
- Extended Pintos to support user programs by implementing system calls, managing the process control block, and separating user threads from kernel threads.
- Implemented virtual memory support in Pintos using paging with swapping, memory management with the buddy system, and page swapping with a **second chance algorithm**.
- Enhanced Pintos with a more efficient file system design, incorporating features like inodes, multi-level index files, directories, and a buffer cache for improved performance.

**Web Learning Management System** | *HTML5/CCS3, Javascript, Flask, SQLite, git* Jan 2021 – Apr 2021

- Created a website that allowed student and teachers to interact, by allowing teachers to post lecture material, assignments and grades, and allowed students to submit assignments and ask questions on a discussion board.
- Engineered a responsive website using **HTML/CSS and JS** in the frontend as well as **Python Flask**, and **SQLite** in the backend.

**Asteroids Game** | *Assembly* Jan 2021 – Apr 2021

- Built a asteroids dodging game in **Assembly**

**Simple Server and Client** | *C* Jan 2021 – Apr 2021

- Employed **signals to interrupt programs** and **sockets** to establish two-way communication between a simple server and client.

**Machine Learning Image Number Classifier** | *C* Jan 2021 – Apr 2021

- Developed a **machine learning** handwritten number classifier using **kNN** with a **97%** accuracy rate.
- Enhanced the machine learning handwritten number classifier by transitioning from kNN to a **decision tree**, reducing running time from **45 minutes to 12 minutes**.
- Implemented **multiprocessing** to accelerate the classification of 10,000 items, reducing processing time from **45 minutes to 5 minutes** by distributing work across **multiple forked processes and communicating via pipes**.
- Utilized **makefiles** for efficient building and compiling of programs.

**Mock Linux Shell** | *Java, SVN* Sep 2020 – Dec 2020

- Developed a **mock (Linux) Shell in Java**, employing various features of **object-oriented programming**.
- Created an **abstract command class** to facilitate **easy development and extension of new commands** (mkdir, ls, cat, redirection, and more).
- Engineered a virtual file system using a **tree structure**.
- Implemented a text-based save file format to preserve the current state of the shell for future loading.
- Engaged in a **2-week agile development cycle** for rapid development and testing using **SVN** to version control.
- Conducted testing of classes using **JUnit test suites and mock classes**.
- Utilized **CRC cards** for brainstorming the design of object-oriented software.

**Budget for the Masses** | *Java, XML, Git, Andriod studio* Feb 2020

- Engineered a budgeting **Andriod app** for the everyday person.
- Built and tested over a 3 day period during the Hack the Valley 4 Hackathon.

**Naive Movie Database** | *C* Jan 2020 – Apr 2020

- Developed a naive database using C, employing **linked lists** and **structs** to manage movie information and reviews.

**Music Sequencer** | *C* Jan 2020 – Apr 2020

- Created a music sequencer with functionalities for storing, organizing, and manipulating musical notes, utilizing a **binary search tree** and supporting music harmonizing.

**Missing Ingredients Swapper** | *C* Jan 2020 – Apr 2020

- Devised an ingredients swapper algorithm utilizing **k-nearest neighbors** and a **graph-based ingredient network**, emphasizing graph concepts and recursion.

**Caesar Cipher** | *C* Jan 2020 – Apr 2020

- Designed a Caesar cipher tool for message encryption and decryption, along with a method to break a Caesar cipher.

**Tissue Sample Simulator** | *C* Jan 2020 – Apr 2020

- Developed a tool simulating tissue samples with various cell types and bacteria using defined functions and data structures.

**Consulate Line Up Tracker** | *C* Jan 2020 – Apr 2020

- Implemented both good and bad versions of a consulate line up, one utilizing a **queue** and the other a **stack** built from scratch.

**Sudoku Solver** | *C* Jan 2020 – Apr 2020

- Designed a sudoku solver using **recursion** capable of solving all solvable 9 by 9 sudoku problems in less than 10ms.

**Tweet Parsing Library** | *Python* Sept 2019 – Dec 2019

- Developed a tweet parsing library to validate tweet content.

**Elevation Map Library** | *Python* Sept 2019 – Dec 2019

- Designed an elevation map library with functionalities such as peak detection and map resolution reduction.

**School Club Recommender** | *Python* Sept 2019 – Dec 2019

- Engineered a club recommendation library utilizing databases of people, their friends, and their club affiliations.

## Academic Experience

**Artificial Intelligence** | *Python*

- Studied the theories and algorithms of Artificial Intelligence. Topics include search, game playing, logical representations and reasoning, planning, reasoning and decision making with uncertainty, computational perception. Assignments provide practical experience of the core topics.
- Looked at many different search methods, like uninformed search methods (DFS, BFS, UCS), and informed search methods (Greedy Search, A*).
- Explored Constraint Satisfaction Problem and how to solve them (Backtracking Search, Cutset Conditioning), and efficiently choose unassigned variables with filtering (Forward Checking, Constraint Propagation, Arc Consistency) and ordering (Minimum Remaining Values, Least Constraining Value).
- Examined adversarial search problems, and how to solve them using minimax values, and how to make it more efficient with Alpha-Beta Pruning, and Depth-limited search. Followed up with hhow to solve adversarial search problems with uncertainty using Expectimax, and Monte Carlo Tree Search.
- Learned how to find optimal policies for simultaneous games, by finding Nash Equilibrium points, solving Markov Decision Processes (Value Iteration, Policy Iteration)
- Acquired techniques to create reinforcement learning agents, with techniques like passive reinforcement learning (Model-based, Model-free), active reinforcement learning (Exploration, Exploitation).
- Learned to use bayesian networks and hidden Markov model to make inferences
- Created a artificial intelligent pac-man agent that explored paths through the mazes, to both reach a particular location and to collect all food efficiently. Used DFS, BFS, UCS, varying the cost function, A* (with many admissible heuristics, and even good heuristics for sub optimal performance, but better speed).
- Engineered a artificial intelligent pac-man agent, that completed the pacman game while avoiding the adversarial ghost. Agents included reflex agents, Minimax (with and without alpha-beta pruning), Expectimax (with different evaluation functions).
- Built artificial intelligent agents to use Markov Decision Processes and reinforcement learning to solve various problems, like grid world, simulated robot controller (Crawler) and Pacman.
- Developed artificial intelligent agents to hunt ghosts down in ghost busters, used bayes nets and factors to infer the location of the ghosts, with variable elimination.

**Computer and Network Security** | *C, Python*

- Studied Public and symmetric key algorithms and their application; key management and certification; authentication protocols; digital signatures and data integrity; secure network and application protocols; application, system and network attacks and defences; intrusion detection and prevention; social engineering attacks; risk assessment and management.
- Worked with different public and symmetric key algorithms like Caesar Ciphers, Substitution Ciphers, Polyalphabetic Ciphers, One Time Pad Ciphers, Stream Ciphers (RC4, Salsa20/Chacha20), Block Ciphers (DES, AES), RSA, ECC. Also exploited different vulnerability like key reuse attacks, known plain text, etc.
- Explored cryptographic hash functions and message authentication codes (MD5 SHA-1, etc), and exploited different vulnerability like hash length extensions. Also used GPG to encrypt and sign messages.
- Hacked networks to conduct different types of attacks, like packet sniffing (with wireshark, and scapy python libaray), malicious gateway, APR spoofing, ICMP Ping flood, TCP-syc flooding, TCP connenction reset, DNS cache poisoning, http sniffer, ssl striping, as well as doing general penetration testing (using nmap, openvas, metasploit).
- Looked at rainbow tables to reverse hashed passwords.
- Hacked systems using different attacks like buffer overflow (Stack Smashing Attack (Branching), Stack Smashing Attack (Shellcode), Adjacent Memory Shellcode), TOCTOU, hidden malware (malware inside a Debian package).
- Hacked web application using various attacks like Incomplete Mediation, SQL injection (SQLi), Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF).
- Wrote a protocol that is similar to TLS version 1.3, that used the Diffie Hellman key exchange algorithm.
- Examined different ways to protect against exploits, like using a nonce, Public key infrasturcture, web of trust, signed certificates, firewalls, Vpns, TOR, salted hash password storage.

**Coding Theory and Cryptography** | *Theory*

- The main problems of coding theory and cryptography are defined. Classic linear and non-linear codes. Error correcting and decoding properties. Cryptanalysis of classical ciphers from substitution to DES and various public key systems [e.g. RSA] and discrete logarithm based systems. Needed mathematical results from number theory, finite fields, and complexity theory are stated.

**Engineering Large Software Systems** | *Python, Git*

- Explored the theory and practice of large-scale software system design, development, and deployment, covering project management, advanced UML, requirements engineering, verification and validation, software architecture, performance modeling and analysis, and formal methods in software engineering.
- Contributed to an open-source Python repository called Langchain, where new tools were developed to enable AI agents to access information from Google services such as Google Jobs, Google Finance, Google Trends, and Google Lens by calling external APIs and parsing the data for the AI agent's consumption.
- Designed and documented an architectural solution for a digital healthcare tracking system, beginning with identifying its functional and non-functional requirements, followed by designing a preliminary system using the C4 model. The design was then analyzed through a risk assessment based on the requirements, and iterative improvements were made to reach a final design.

**Algorithm Design and Analysis** | *Theory*

- Explored various algorithm design techniques including divide-and-conquer, greedy strategies, dynamic programming, linear programming, randomization, and others.
- Studied greedy algorithms such as the interval scheduling problem, minimum lateness scheduling, Dijkstra's algorithm, and Huffman codes.
- Learned divide-and-conquer algorithms including merge sort, binary search, Karatsuba's multiplication, Strassen's matrix multiplication, closest pair of points, and order statistics, along with their running time analysis.
- Explored dynamic programming algorithms such as the longest increasing subsequence, weighted interval scheduling, optimal sequence alignment, Optimal Binary Search Tree, and 0-1 knapsack.
- Studied different shortest path algorithms including Bellman-Ford, Floyd-Warshall, and Johnson's algorithm.
- Worked with max flow/min cut algorithms (e.g., Ford-Fulkerson) and their applications like bipartite graph matching, minimum vertex covers in bipartite graphs, and finding the maximum cardinality set of edge-disjoint paths.
- Solved problems using linear programming techniques such as the Diet problem, Profit maximization, max flow, min cost flow, and linear regression. Also examined integer linear programming problems like facility location.
- Explored approximation algorithms for hard-to-solve problems like vertex cover, set cover, minimum makespan, k-centers, and the knapsack problem.

**Programming on the Web** | *HTML5/CSS3, Javascript, React, Express, MongoDB, Docker, Web Sockets, Git*

- Introduced to software development on the web, covering operational concepts of the internet and the web, static and dynamic client content, dynamically served content, n-tiered architectures, web development processes, and web security.
- Learned and utilized JavaScript (including concepts like Asynchronism, Event Loop, and Web Workers) to develop dynamic HTML and CSS websites and applications.
- Acquired skills in building web APIs following the REST framework, enabling data transfer between frontend and backend, including file handling.
- Worked with cookies, sessions, and various web authentication methods (Local, Token Based, Third Party), and implemented security measures against common attacks such as Cryptographic Failure, Broken Access Control, Injections (SQL/XSS), and Security Misconfiguration (CSRF/CORS).
- Explored different communication methods between applications, including long polling, short polling, web sockets, and WebRTC.
- Deployed real applications to the internet using technologies such as Docker, VMs, domains, reverse proxies, certificates, web packing, HTTP/2, HTTP/3, progressive web apps, and CDNs.
- Studied methods for internationalizing apps (i18n) with specific localization (L10n).
- Explored web advertising, analytics, tracking, and fraud detection techniques.
- Developed an online web gallery allowing users to upload image collections and comment on others' images, utilizing HTML5, CSS3, and JavaScript.
- Created a drag and drop website builder using React and ExpressJS, enabling multiple users to collaborate on the same site, chat using web sockets, and share videos and audio with WebRTC.
- Deployed the website on a Google Cloud virtual machine using Docker for frontend and backend images, with an Nginx reverse proxy for hosting both on the same machine, and automated the process using a shell script.

**Social Impact of Information Technology** | *Theory*

- The trade-offs between benefits and risks to society of information systems, and related issues in ethics and public policy. Topics will include safety-critical software; invasion of privacy; computer-based crime; the social effects of an always-online life; and professional ethics in the software industry. There will be an emphasis on current events relating to these topics.

**Introduction to Databases** | *Java, MySQL*

- Introduced to database management systems, covering the relational data model, relational algebra, and the SQL query language for querying and updating databases, along with application programming with SQL.
- Explored integrity constraints, normal forms, and database design principles, including elements of database system technology such as query processing and transaction management.
- Studied the entity-relationship model/diagrams and different types of relationships like one-to-one, many-to-one, and many-to-many relationships.
- Learned to convert entity-relationship diagrams into relational models, allowing for relational algebra operations and computation of functional dependencies, design anomalies, and normalization into Boyce-Codd Normal Form (BCNF) and Third Normal Form (3NF).
- Worked with SQL to implement relational models into databases for querying and manipulation.
- Developed a replica database of Airbnb in MySQL, enabling users to add listings with multiple amenities for any dates and allowing other users to rent out the listings, with the relation normalized into 3NF and accessible via Java.
- Designed a mock Twitter database allowing users to create tweets and comments, like other users' tweets and comments, and retweet others' tweets.

**Human-Computer Interaction** | *Figma, Balsamiq Wireframes*

- Explored the field of Human-Computer Interaction (HCI), focusing on guidelines, principles, methodologies, and tools for analyzing, designing, and evaluating user interfaces.
- Learned to design software with a primary emphasis on UI/UX, considering factors like universal usability, the 8 golden rules for interface design, and user-centered design principles.
- Studied how to establish requirements using various data gathering methods such as interviews and surveys to understand and map user interactions, utilizing tools like Hierarchical Task Analysis (HTA) to make low-level designs with wireframing, creating interactive prototypes, and evaluating the design through testing.
- Built an interactive prototype UI using Figma for a task management app allowing users to create groups and share tasks on the board.
- Conducted user testing on the UI through surveys and semi-structured interviews to identify strengths and weaknesses, iterated on the design based on feedback to enhance usability and user experience.

**Introduction to Software Engineering** | *React, MongoDB, ExpressJS, NodeJS, Jira*

- Explored software development methodologies with a focus on agile methods suitable for rapidly moving projects, covering basic software development infrastructure, requirements elicitation and tracking, prototyping, project management, basic UML, software architecture, design patterns, and testing.
- Studied the software development lifecycle and various models including waterfall, V-shape, and agile, with a detailed examination of scrum elements such as product backlog, sprint backlog, sprint planning, daily scrum, sprint retrospective, sprint review meeting, burnup and burndown charts, pair programming, and test-driven development.
- Learned how to scope projects by creating user stories, use cases, personas, acceptance criteria, and estimating stories.
- Explored REST API design and database types including SQL, NoSQL, and GraphQL.
- Investigated different software architectures such as Multi-Tier architecture, Multi-Layer architecture, MVC, Microservices, Monolithic, and Event Driven, and their design in UML considering non-functional requirements like scalability, performance, and availability, using use case diagrams, sequence diagrams, activity diagrams, and state chart diagrams.
- Studied design principles like SOLID and design patterns such as singleton, factory method, builder, and strategy design pattern.
- Explored Git Flow including merging, rebasing, and PRs, and release management.
- Learned about continuous integration and continuous deployment, utilizing Docker to componentize different aspects of the system.
- Collaborated with a startup founder to build a MERN platform facilitating interaction between students and companies for internships and competitions.
- Developed a short polling chat application within the platform for users/companies interaction.
- Automated the deployment process using Git actions to build Docker images and push them to Docker Hub.

**Introduction to Machine Learning and Data Mining** | *Python*

- Explored various methods for automated learning of relationships from empirical data, including classification and regression techniques such as nearest neighbor methods, decision trees, linear and non-linear models, class-conditional models, neural networks, and Bayesian methods.
- Investigated clustering algorithms and dimensionality reduction techniques, as well as model selection and the problems of overfitting and assessing accuracy, especially with large databases.
- Studied linear regression in one dimension, multiple dimensions, and multiple output dimensions, as well as non-linear regression with polynomial and radial basis functions.
- Explored the bias-variance trade-off and regularization to address overfitting and underfitting, compared generative vs. discriminative models.
- Implemented clustering with k-means, hierarchical k-means, k-means++, product quantization, and Gaussian mixture models.
- Explored classification techniques including regression, k-nearest neighbors, decision trees, class conditionals, naive Bayes, and logistic regression, along with testing different models with cross-validation.
- Investigated AdaBoost with weak classifiers like decision stumps, Support Vector Machines with the Lagrangian and kernel trick, and Principal Component Analysis for dimensionality reduction.
- Studied neural networks and the backpropagation algorithm.
- Developed and compared models for university admission prediction using linear regression and k-means++, including clustering the data before applying linear regression.
- Applied radial basis functions to restore corrupted images in a painter program and wrote a basic logistic regression script to classify objects as oranges based on width and height.
- Implemented and evaluated classifiers for labeling news articles into five categories using Naive Bayes, Gaussian class conditional, and k-nearest neighbors classifiers.
- Developed and evaluated an SVM classifier and an SVM classifier with PCA to predict whether the emotion associated with a human audio signal is positive or negative using Mel spectrogram and chromagram features.

**Analysis of Numerical Algorithms for Computational Mathematics** | *Theory, Matlab*

- Explored the efficiency, accuracy, and reliability of numerical algorithms for various models including least squares, non-linear equations, optimization, quadrature, and systems of ordinary differential equations.
- Investigated methods for solving over and under-determined linear systems, including solving normal equations, QR factorization using Householder reflections, and Singular Value Decomposition.
- Solved systems of non-linear equations by transforming root-finding problems into fixed-point problems, with a focus on Newton's Method and its efficiency analysis.
- Examined various approximation and interpolation techniques, such as the Vandermonde method, Lagrange Basis (Lagrange Hermite Interpolation), Newton's polynomial for general interpolations, and piecewise polynomial interpolation. Analyzed the error in each interpolation method and methods to control it, including the use of Chebyshev points.
- Analyzed different methods for numerical integration (quadrature), including standard forms, midpoint rule, trapezoidal rule, Simpson's rule, and composite rules, evaluating their precision and error characteristics.
- Studied methods for solving Initial Value Problems for Ordinary Differential Equations, including Forward Euler's method and Backward Euler's method. Explored stability analysis using Gershgorin's Circle Theorem, assessing accuracy, stability, and region of stability.

**Introduction to Numerical Algorithms for Computational Mathematics** | *Theory, Matlab*

- Studied computational methods for problem-solving in linear algebra, non-linear equations, approximation, and integration.
- Explored floating-point arithmetic, including concepts of rounding error, error propagation, conditioning, and stability of functions.
- Examined techniques for solving linear systems of equations, such as LU factorization with row pivoting, and iterative improvement methods.
- Investigated methods for solving non-linear equations, including approaches involving fixed-point problems, convergence analysis, and rate of convergence improvement using Newton's method, secant method, bisection method, and hybrid methods.
- Explored approximation techniques including interpolation methods such as Vandermonde method, Lagrange basis, Newton (Divide Difference) Basis, and Newton polynomial, along with error determination in polynomial interpolation.
- Studied numerical integration (quadrature) and common rules such as the midpoint rule, trapezoidal rule, and Simpson's rule, along with their respective errors.

**Principles of Programming Languages** | *C/C++, Java, Python, Scheme, ML, Haskell, Prolog*

- Explored programming paradigms including procedural (e.g., C, Java, Python), functional (e.g., Scheme, ML, Haskell), and logic programming (e.g., Prolog).
- Utilized higher-order functions such as map, filter, fold, and others in programming tasks.
- Developed functional programs employing techniques like pattern matching, tail-recursion, list comprehension, lazy evaluation, and closures.
- Gained practical experience in various programming paradigms through hands-on work and study.

**Computability and Computational Complexity** | *Theory*

- Introduction to the theory of computability: Turing machines, Church's thesis, computable and non-computable functions, recursive and recursively enumerable sets, reducibility. Introduction to complexity theory: models of computation, P, NP, polynomial time reducibility, NP-completeness, further topics in complexity theory.

**Operating Systems** | *C*

- Studied principles of operating systems, including its roles as a control program and resource allocator, and concepts such as processes, concurrency problems, synchronization, mutual exclusion, and deadlock.
- Enhanced the basic Pintos OS by incorporating features like context switching between threads, support for user programs, virtual memory management, and implementation of a file system.
- Improved Pintos to efficiently handle multithreading by managing interrupts, implementing context switching with a Multilevel Feedback Queue scheduler, and ensuring proper synchronization using locks and semaphores to avoid race conditions.
- Extended Pintos to support user programs by implementing system calls, managing the process control block, and separating user threads from kernel threads.
- Implemented virtual memory support in Pintos using paging with swapping, memory management with the buddy system, and page swapping with a second chance algorithm.
- Enhanced Pintos with a more efficient file system design, incorporating features like inodes, multi-level index files, directories, and a buffer cache for improved performance.
- Gained knowledge of virtual machines, Docker containers, distributed systems, and remote procedure calls through study and practical experience.

**Introduction to Databases and Web Applications** | *HTML5/CCS3, Javascript, Flask, SQLite, git*

- Developed a web platform facilitating interaction between students and teachers, enabling teachers to post lecture material, assignments, and grades, while allowing students to submit assignments and engage in discussions.
- Designed and implemented a responsive website using HTML/CSS for the frontend and Python Flask framework with SQLite for the backend.
- Utilized CSS flexbox to implement various UI features, ensuring responsiveness across different devices.
- Acquired proficiency in basic relational algebra and SQL for database management.
- Employed GitHub as a version control and collaboration tool to manage project changes and facilitate teamwork.

**Design and Analysis of Data Structures** | *C*

- Gained understanding of asymptotic complexity through big O, big Omega, and big Theta notations, along with amortized sequence complexity
- Explored advanced data structures including AVL trees, priority queues using Heaps, dynamic arrays, Disjoint Sets, Fibonacci Heaps, Hashing, B-trees, and Bloom Filters
- Learned advanced graph algorithms and their complexities, such as BFS, DFS, Kruskal's algorithm, Prim's algorithm, Dijkstra's algorithm, and Kosaraju's Strongly Connected Components
- Designed, analyzed, implemented, and compared efficient data structures for common abstract data types like; priority queues: Heaps and mergeable heaps; Dictionaries: Balanced binary search trees, B-trees, hashing; Amortization: Data structures for managing dynamic tables and disjoint sets; Data structures for representing graphs and performing graph searches.

**Computer Organization** | *Assembly*

- Learned principles of the design and operation of digital computers. Binary data representation and manipulation, Boolean logic, components of computer systems, memory technology, peripherals, structure of a CPU, assembly languages, instruction execution, and addressing techniques. There are a number of laboratory periods in which we conducted experiments with digital logic circuits.

**Software Tools and Systems Programming** | *C*

- Acquired software techniques in a Unix-style environment, utilizing scripting languages and the C programming language.
- Covered core topics such as creating and using software tools, pipes and filters, file processing, shell programming, processes and multithreading, system calls, signals, and basic network programming with sockets.
- Developed a machine learning handwritten number classifier using kNN with a 97% accuracy rate.
- Enhanced the machine learning handwritten number classifier by transitioning from kNN to a decision tree, reducing running time from 45 minutes to 12 minutes.
- Implemented multiprocessing to accelerate the classification of 10,000 items, reducing processing time from 45 minutes to 5 minutes by distributing work across multiple processes and communicating via pipes.
- Utilized makefiles for efficient building and compiling of programs.
- Employed signals to interrupt programs and sockets to establish two-way communication between a simple server and client.

**Introduction to the Theory of Computation** | *Theory*

- Studied mathematical, mathematical logic, correctness proofs for iterative and recursive algorithms, solutions of linear and divide-and-conquer recurrences, introduction to automata and formal languages.
- Learned about finite state automata, regular expressions, context-free grammars, and pushdown automata.

**Software Design** | *Java*

- Studied software design and development principles, methods, and tools using a statically typed object-oriented language, Java.
- Covered topics including version control, build management, unit testing, refactoring, object-oriented design and development, design patterns, and advanced IDE usage.
- Developed a mock (Linux) Shell in Java, employing various features of object-oriented programming.
- Created an abstract command class to facilitate easy development and extension, along with a mock filesystem using a tree structure.
- Implemented a text-based save file format to preserve the current state of the shell for future loading.
- Engaged in a 2-week agile development cycle for rapid development and testing.
- Conducted testing of classes using JUnit test suites and mock classes.
- Utilized CRC cards for brainstorming the design of object-oriented software.
- Acquired knowledge of various design patterns and their implementation, including Singleton, Builder, Decorator, Iterator, Observer, and Dependency Injection.
- Explored different software development processes such as Agile, Incremental, and Waterfall.

**Introduction to Computer Science II** | *C*

- Studied abstract data types and data structures, including linked data structures, object-oriented programming, encapsulation, recursion, and information-hiding.
- Examined testing, specifications, and efficiency analysis of programs.
- Developed a naïve database using C, employing linked lists and structs to manage movie information and reviews.
- Created a music sequencer with functionalities for storing, organizing, and manipulating musical notes, utilizing a binary search tree and supporting music harmonizing.
- Devised an ingredients swapper algorithm utilizing k-nearest neighbors and a graph-based ingredient network, emphasizing graph concepts and recursion.
- Implemented basic pathfinding on a graph using breadth-first search and depth-first search algorithms.
- Designed a Caesar cipher tool for message encryption and decryption, along with a method to break a Caesar cipher.
- Developed a tool simulating tissue samples with various cell types and bacteria using defined functions and data structures.
- Implemented both good and bad versions of a consulate line up, one utilizing a queue and the other a stack built from scratch.
- Designed a sudoku solver using recursion capable of solving all solvable 9 by 9 sudoku problems in less than 10ms.
- Constructed various data structures (Lists, Binary Trees, Graphs, etc.) as projects in C
- Explored memory models on a low level and their utilization by different data types
- Analyzed program efficiency and big O notation by studying various sorting algorithms and data structures.
- Examined principles of designing and building good software, focusing on properties such as modularity, reusability, extendibility, maintainability, correctness, efficiency, openness, privacy, and security.

**Introduction to Computer Science I** | *Python*

- Acquired foundational knowledge in programming including, elementary data types, statements, control flow, functions, classes, objects, and methods.
- Familiarity with lists, searching, sorting, and basic complexity.
- Developed a tweet parsing library to validate tweet content.
- Designed an elevation map library with functionalities such as peak detection and map resolution reduction.
- Engineered a club recommendation library utilizing databases of people, their friends, and their club affiliations.

## Technical Skills

**Languages**: Java, Python, C, JavaScript, Typescript, HTML/CSS
**Storage**: MySQL, SQLite, MongoDB, Firebase Store, Supabase
**Frameworks**: React, NodeJS, ExpressJS, NestJS, NextJS, Flask, JUnit, Jest
**Developer Tools**: Git, GitHub Actions, Docker, Google Cloud Platform, VS Code, Visual Studio, Jupiter Notebook, Eclipse
**Libraries**: Axios, Styled-Components, Ethers, Redux, Recharts, Socketio, Pandas, NumPy, Matplotlib, Chai, Mocha, Hardhat