Mobassera Zaman
608-3433 Rue Durocher, Montréal, QC H2X 2C8, +1 438 924 1770 | mobassera.jemi@gmail.com
GitHub: https://github.com/mobasserazaman

EDUCATION & QUALIFICATIONS

McGill University **Fall 2017 – Winter 2021**

Bachelor of Science - Software Engineering

Relevant Courses: Algorithms and Data Structures, Discrete Structures, Operating Systems, Concurrent Programming, Machine Learning, Computer Networks, Computer Vision, Programming Languages and Paradigms, Basics of Web Development, Introduction to C++

SKILLS

Technical/Computer:

- **Programming Languages:** Java, JavaScript, Python, C, C++, C#, SQL, PHP, HTML, CSS, OCaml
- **Web Frameworks and Libraries**: Bootstrap, ReactJS, Django
- **Python Libraries:** TensorFlow, Pandas, NumPy
- **Development Tools:** Visual Studio Code, Eclipse, Sublime Text, Atom
- **Publishing Software:** Microsoft SharePoint, Word, Excel (Macros), PowerPoint, Access, Outlook, EndNote
- **Operating Systems:** Linux, Windows OS, MacOS
- **Project Management and Communication**: Basecamp, Slack
- **Miscellaneous**: GitHub, Unity, LaTeX, TravisCI, Maven, Gradle, Heroku

Behavioral: Teamwork, Adaptability, Problem-solving, Leadership, Time management.

Language: Fluent in English and Bengali

EXPERIENCE AND PROJECTS

McGill School of Computer Science Website **Fall 2020**

- Developed a complete website with over 70% of the code base built from scratch.
- Designed a responsive and interactive front-end using HTML5, CSS and JS as a minimum.
- Set up a working back-end using PHP to support the dynamic update of contents and MySQL to build an on-server database, and later to connect to it.
- Utilized XAMPP to create a local web server on the computer to fast-track development and make local testing of the codes feasible.
- Used SSH to transfer files to the McGill SOCS web server.
- Collaborated with teammates to choose technology stack, divide tasks, design, and document the project.

Simplified Link State Routing Protocol with Java Socket Programming **Winter 2020**

- Designed a pure user-space program which simulates the major functionalities of a routing device running a simplified Link State Routing protocol.
- Implemented a router class and mapped the ‘Process IP’ and ‘Process Port’ of each program instance to a ‘simulated IP address’ attribute used to identify the router in the simulated network space.
- Added a Link State Database which is a map from the router's IP address to the link state description which is originated by the corresponding router and Link State Advertisement structures that summarize the latest link state information of the router.
- Developed a command line-based terminal interface for the router to allow the user to input the following commands: attach, start, connect, disconnect, detect, neighbours, quit. Some of these commands trigger Link, State Database synchronization and update.
- Built a weighted graph representing the topology of the network and used Dijkstra’s algorithm to find the shortest path between routers.
- Implemented a message class that defines the message format transmission among routers.

Logistic Regression and Naive Bayes **Winter 2020**

- Implemented two classification techniques in Python —logistic regression and naive Bayes from scratch on four distinct datasets and compared the accuracy of both classification techniques on the four datasets.
- Cleaned the datasets by removing any data with missing or malformed features and used one-hot encoding for categorical variables.

- Computed basic statistics such as distributions of the positive vs. negative classes, distributions of some of the numerical features, correlations between the features etcetera to get a better understanding of the data.
- Implemented a script to run k-fold cross validation.
- Compared the accuracy of naive Bayes and logistic regression on the four datasets.
- Compared the accuracy of the two models as a function of the size of the dataset (by controlling the training size).
- The team (3 people) collectively worked to prepare the processing, classification and evaluation code and the project write-up.

Classification of Image Data **Winter 2020**

- Developed models to classify image data (used the CIFAR-10 dataset with the default test and train partitions).
- Implemented Multilayer perceptron from scratch using backpropagation and the mini-batch gradient descent algorithm used (e.g., SGD).
- Compared and reported the test and train performance of the above two models as a function of training epochs.
- Used Python libraries to extract features and preprocess the data, evaluate model, and to tune the hyper-parameters, etc.
- Prepared a write-up (analysis and conclusion) on the results.

Operating System using C **Winter 2019**

- Built an OS Shell which accepts commands such as help, quit, set VAR STRING, print VAR, exec prog1 prog2 prog3, run SCRIPT.TXT etcetera.
- Implemented the following data structures: CPU, PCB, ready queue, and RAM and the following algorithms: scheduler, task switch, and basic memory management.
- Added OS Boot Sequence (Prepare the Backing Store and Prepare RAM for paging) to create necessary OS structures.
- Added Memory Manager (Launcher, PCB modifications, Page Fault, Task Switch) to handle memory allocation for processes.
- Implemented Partition Table, File Allocation Table, Block buffer along with functions such as partition, mount, read/writeBlock etcetera to add some of the functionalities of a File System to the OS.
- Wrote a public IOScheduler function to handle up to 10 requests from exec programs.

Multiplayer Flash Point Game **Fall 2018 Winter 2019**

- Designed and developed a multiplayer board game using C# on the gaming engine Unity.
- Designed an advanced User Interface using the tools offered by Unity to provide multiplayer gaming features such as context sensitive help and messaging between players and such.
- Client-server framework logic was developed to allow connections to players online using C# on Unity.
- Managed a group of 5 designers to design, develop and document the entire project.
- Designed test cases and ran unit tests to validate the proper and desired functionality of the game.
- Learned to use Git protocol (fetch/pull/push/merge etc.) on GITHUB for version controlling the software during design phase.

AWARDS

- The Daily Star Award for Exceptional Performance in GCE Ordinary and Advanced levels. **2014, 2016**

VOLUNTEER EXPERIENCES

- Helped organize Interschool Basketball Tournament **2013**
- Helped organize the Disciplinary Month activities. **2014**

INTERESTS

Programming, Travelling, Socializing, Music, Soccer, Chess, Video Games, Dancing