HAMZA ARSHAD

DEPARTMENT OF SYSTEMS DESIGN ENGINEERING · SEPT 2017 – APRIL 2022
github.com/hamzanadeveloper | hamzaarshad.com | hamza.n.arshadwork@gmail.com | 289-941-4545

SKILLS

- Proficiency with **JavaScript** (**ES6+**) and experience with **Ruby**, **Python**, **C**, **PHP**, **HTML5**, and **CSS3**

FRAMEWORKS

- Front-end: Experience with **React** & **Redux**, **Angular**, **Vue**, **D3**, **Svelte**, **PhantomJS**, and **jQuery**
- Back-end: Built **APIs** using **REST** and **WebSockets** with **Node (Express)**, **Ruby on Rails**, and **Flask**
- Mobile: Built cross-platform mobile applications using **React-Native** and **Expo**
- Testing & Automation: Employed **test-driven development** & **continuous integration** with **Jest** and **Drone**
- Server & Tools: Experience with **AWS**, **Firebase**, **Docker**, **RabbitMQ**, **Postgres** and **MongoDB**

EDUCATION

UNIVERSITY OF WATERLOO · 4.0 cGPA

- Bachelor of Applied Science in Biomedical Engineering with option in Artificial Intelligence (3B)
- Courses: Pattern Recognition (**Python**), Digital Computation (**C#**), Data Structures & Algorithms (**C++**)

RELEVANT EXPERIENCE

SOFTWARE ENGINEERING INTERN – APRIL 2020 – SEPT 2020 — University of Toronto

- Refactored a monolithic app to a **Dockerized microservice-based app** using **Express** and **RabbitMQ**
- Migrated a **RESTful** HTTP request architecture to **WebSockets** using **SocketIO** to enable real-time interactions
- Enhanced **UI/UX** of app by designing with **MaterialUI** and improving interactivity with **React Hooks** & **Redux**
- Assisted in implementing a **CNN model** using **OpenCV** and **PyTorch** to predict surgery probability of kidneys
- Implemented **blob storage**, accessible in a volume between microservices, to store ultrasounds of kidneys

SOFTWARE ENGINEERING INTERN – SEPT 2019 – DEC 2019 — The Hospital for Sick Children (SickKids)

- Made a progressive web application using **AWS**, **React**, **Feathers**, and **Docker** to let clinicians contact patients
- Reduced bundle size in **Webpack** and implemented Role-Based Access Control with **Redux** and **React-Router**
- Built API test suites using **Jest** and **QUnitJS** to assess the security of RareConnect, a social media platform
- Developed an iOS/Android application to display push notifications using **React-Native**, **Expo**, and **Firebase**
- Produced web applications for researchers using **AWS EC2** instances, **Pinpoint**, **SNS** and **SES**, and **Docker**

FULL STACK DEVELOPER – JAN 2019 – APRIL 2019 — HealthIM

- Configured API routes for web, mobile, and desktop clients on **Ruby on Rails** back-end
- Used modern-web technologies such as **Angular**, **Node** and **Phantom** to create seamless desktop clients
- Improved an existing dashboard using data-visualization technologies, such as **D3**, **ChartJS**, and **Highcharts**
- Presented to hospitals, police services, and relevant stakeholders regarding product lifecycle and requirements
- Worked in a cross-functional agile team, using **JIRA** and **Taskworld** to document progress

HAMZA ARSHAD

DEPARTMENT OF SYSTEMS DESIGN ENGINEERING · SEPT 2017 – APRIL 2022

github.com/hamzanadeveloper | hamzaarshad.com | hamza.n.arshadwork@gmail.com | 289-941-4545

PROJECTS

FACIAL RECOGNITION AUTHENTICATION – JUNE 2020

- Replaced resetting password workflow on a web application with facial recognition authentication
- Created a seamless registration and sign-in workflow, leveraging **React** and **Semantic UI** on the client-side
- Implemented a **Tiny Yolo V2 CNN** for facial detection on the client-side, and facial recognition on the API
- Built a **Dockerized REST API** using **Express**, and stored relevant user information using **PostgreSQL**

INSIGHT PROTOTYPE – OCT 2019

github.com/hamzanadeveloper/aws-feathers-setup

- Developed a web application boilerplate that enabled physicians to report patient outcomes to their clients
- Produced both a web and mobile responsive interface using **MaterialUI**, **React**, and **React Router**
- **REST API** built using **Express**, and communication to the client was handled via **AWS** services and **Docker**
- Built on an **AWS EC2** instance, and used services, including but not limited to, **Pinpoint**, **SNS**, and **SES**

MNIST DIGIT RECOGNIZER – MAY 2020

github.com/hamzanadeveloper/digits-recognizer

- Captured user input via a 224x224 canvas, which was then processed and downsampled to a 28x28 image.
- Built a 6-layer CNN with a 96% accuracy using the MNIST dataset, leveraging **TensorflowJS** for digit recognition
- Utilized **React** and **MaterialUI** to enhance the **UI/UX** of the application, and ran the model client-side

ENGAGE PROJECT– NOV 2019

github.com/hamzanadeveloper/reactnativepush-native

- Investigated methods to re-engage users in a clinical setting and pursued a cross-platform mobile application
- Used **React-Native** and **Expo Client** to quickly build an **Android/iOS** mobile application
- Incorporated **Google's Firebase API** to facilitate pushing out notifications using **FCM** and **APN** device tokens
- Produced a prototype and presented my findings to the **Centre for Computational Medicine**

ANALYTICS – JAN 2019

- Used **Angular**, **Node, D3**, and **jQuery** to generate reports that would amass statistics about and organization
- Worked on a **Rails** back-end to retrieve data from **PostgreSQL** and used sorting algorithms to present the data
- Made use of JavaScript's asynchrony and promises to render charts in a headless browser using **PhantomJS**
- Created a rake task paired with a **CRON** job to automate mailing to clients in the application
- Implemented **Node-crypto**, which decrypted identifiable information using **RSA-AES** encryption

CASE MANAGEMENT – APRIL 2019

- Used **Angular** and **CSS3** to produce a seamless interface to show patient history to hospital administration
- Encrypted all identifiable data in-transit with 4096-bit **RSA-AES encryption** and stored data in **PostgreSQL**
- Created and configured routes in **Ruby on Rails** to retrieve and post data to the database
- Upon completion of the application, we began to rebuild the application in **React**