# Kinshuk Phalke

+1-(404)-519-7018 | phalkekinshuk@gatech.edu | linkedIn/kinshukphalke | github/xinslu | kinshukphalke.com

## EDUCATION

**GEORGIA INSTITUTE OF TECHNOLOGY** Atlanta, GA | Aug 2021 - May 2024
Bachelors of Science in Computer Science **GPA:** 4.0
**Coursework**: Data Structure and Algorithms, Honors Discrete Mat
**Clubs**: GreyHat (CyberSec Club), dependently-typed (Programming Language Club)

## WORK EXPERIENCE

**NCR CORPORATION** | Software Engineering Intern Atlanta, GA | May 2022 - Aug 2022

- Worked in the Retail innovation team in building a micro-service in Go to enable Cart service enabling shopping, checkout, transactions, taxes and payments.
- Made API endpoints, Integrated with other APIs for additional services like taxes, prices, location and item attributes.
- Modelled data and created a CI/CD pipeline that performs a series of checks and publishes the Kubernetes Cluster onto Google Cloud Platform.
- Wrote a Bash Script with 12 presets to help change environment variables for different stages in production, enable simple deployments, creating and updating internal CA Certificates and managing kubectl pods and deployments.

**NOTITIA** | Software Engineer Atlanta, GA | Nov 2021 - Present

- Working as full stack engineer in building a MVP for a startup at Georgia Tech.
- Built a LDA model in Python and worked with tuning UMAP parameters and implemented a clustering algorithm using FAISS. Worked on UI of a WordCloud using React Flow.
- Optimized the front-end by 60% and back-end by 40%.

## PROJECTS

**SPEECHTOCODE** Python, ReactJS, Node.js
Built a meta language parser enabling you to speak english to create a class based abstraction, which is then easily translated to the language selected. Added support for javaScript, Python and Java with near full language features. Worked on designing the frontend and connecting the parser to the frontend using a flask server.

**DEUSCL** Rust, Common Lisp, Yew Framework
A memory-safe REPL built in Rust. Built a handwritten recursive descent parser and a visitor pattern dependent evaluator. Added a basic type inference, type checking and error handling system. Added dynamic evaluation of functional environment and parameters. Wrote a simple garbage collector and environment cleaner. Building a frontend in Yew Framework in Rust to enable users to use a web based simulation of REPL.

**PUBLIXMON** MongoDB, Express.JS, React Native, Node.JS, Flask, Node.js
Publixmon is an app to boost retail engagement. Built a smart contract allowing users to mint NFTs based on Order History of each user. Integrated with the NCR BSP layer to pull order details, catalog and inventory details. Built and tested mining NFTs on the Ganache Blockchain and built a node.js server to facilitate blockchain actions. Project Won the First Place at the NCR API Challenge at HackGT 8.

**MENTAL HEALTH CHAT** Redis, Docker, MongoDB, Express.JS, ReactJS, Node.JS, Socket.IO
A timed chat for users to talk about mental health issues. Built a login platform using firebase and OAuth, utilized redis cache inorder to create cache to store the messages for 6 hours before deletion, served requests throught a REST Node.js server, utilized Socket.IO to create pubsub connections between clients and utilized Redux as a centralized theme store per user.

**HACK-A-LANG** Python
Worked with a team of 3 for the ”hack-a-lang” event, to make the python implementation of the Lox Learning Language, complete with an implementation of dynamic functional environments, type checking, error handling, recursive descent parser and a visitor patter dependent evaluator. Also, spoke at the event about the evaluator, visitor patterns, environment creation and variable handing. Also, created exercise for the participants to work on to increase their knowledge.

**ENHANCE** Python, Tensorflow, Keras
Created a Res-Net Implementation of the Super Resolution GAN Paper (https://arxiv.org/pdf/1609.04802.pdf) for Single Image Super Resolution. I worked with tensorflow in creating a Residual Blocks and Skip Connections and used the VGG-19 Model to calculate the Content Loss of the Model. The model achieved perceptible improvements in image quality on the DIV2K dataset with a downscale factor 4.

## SKILLS

**Languages**: Python, Rust, Go, JavaScript, Java, C, C++, C#, , Solidity
**Full-Stack**: ReactJS, Express.js, Node.js, Flutter, Ionic, React Native, Socket.IO, Redux, Web3.js, Yew, Tauri
**Machine Learning**: NumPy, Scikit-learn, Open AI Gym, Matplotlib, Pandas, Keras
**Technologies**: Git, Docker, Postman, LaTeX, SQL, MongoDB, Redis, Kubernetes