Edson S. Cortes Rivera

https://www.edcortes.dev | (443) 766 – 6576 | edcortes@terpmail.umd.edu

Education:

University of Maryland, College Park, MD
Bachelor of Science, Computer Science **Graduated: December 2022**
Minor: General Business
UG Cumulative GPA: 3.54

Skills:

Languages: Python, Java, C/C++, JavaScript, OCAML, Kotlin, SQL, HTML, CSS, Ruby.
Technologies: Git, React, PostgreSQL, FastAPI, AWS Amplify, Route 53, Lambda, EC2, and Cognito.

Experience:

Student Sustainability Summit | *Software Engineer* | *Fall 2022 - present*

- Joined the software engineering team at the Student Sustainability Summit at UMD and worked on the organization's software. This included the official website https://www.studentsustainabilitysummit.org and the official organization's application. Designed and built the front end of the apps.

Projects:

AI Inverse Art Generator | *AI art generator* | *Python and React*

• AI art image generator that takes in a prompt, inverses that prompt (using antonyms), and displays an AI generated image of that inversed prompt. For example, entering "A big black dog" would return an AI generated image of a "Small white dog". OpenAI's DALL-E is used to generate images. The front-end website was coded using React and the backend API was coded using Python and FastAPI.

Rank My Store | *Android Phone Application* | Kotlin

• Android application made using the coding language, Kotlin in Android Studio. Collaborated with two other Kotlin developers to create a social media app that allowed users to rate, rank, and share pictures of grocery stores in their area. Google Firebase was used to handle user authentication and for Database storage. Google Places API was used to get the user's location and nearby grocery stores. The ranking and store rating is based on healthy fruits and vegetables available at the given grocery store.

LAVI | *Language Agnostic Vulnerability Identifier* | Python

• CLI tool that allowed users to pass in a package along with its dependencies and output the package's vulnerabilities and exposures (CVE'S) associated with those dependencies and sub-dependencies. The CLI would communicate to an API via HTTP endpoints. The API would then query the LAVI database for CVE data associated with passed in package dependencies. This also included a LAVA (Language Agnostic Vulnerability Analysis) feature to help users get statistics on CVE's and Repositories such as Pip, NPM, and Golang. The queries that LAVA answered were such as, "how many CVE's exist in a certain repo".