Ryosuke Urawa

4th year, Computer Science

UBC Science Co-op
T: 604.822.9677 | science.coop@ubc.ca
www.sciencecoop.ubc.ca

Vancouver, Canada 778-321-9562 |ryosukeurawa@gmail.com| LinkedIn | github.com/Ryo-Ura

Education

University of British Columbia

Bachelor of Science in Computer Science (GPA: 3.70)

Langara College

Associate of Science in General Science (GPA: 4.04) Langara Student Union International Education Scholarship, 2 Dean's honor role

Technical Skills

Languages: Python, JavaScript, Java, C++, C, Dart, Ruby **Frameworks**: Flutter, React, Node, Rails, Pytest, Redux

Developer Tools: Git, Docker, PostgreSQL, NoSQL, GraphQL, datadog, Postman

Work / Volunteer Experience

Mobile Integration Intern (Fulltime), Rivian, Vancouver Jan 2023-Present

- Presented the automation test plan for coming new feature and implemented the automation by release date
- Updated the existing automation structure and reduced execution time about 45%
- Implemented a logic to run a pipeline on a certain test bench based on the given runner tag
- Presented and did a knowledge share about benefit of moving from polling to subscription, and implemented a test to run subscription in the background and validate the streamed output

System Engineer Intern (Fulltime), Next, Japan May 2022-December 2022

- Created development environment using Docker, and created dockerfile that supports M1 mac tip
- Implemented keyword search functionality in backend to find a specific power plant in Ruby on Rails
- Wrote a test and craeted test data table on PostgreSQL to test and debug the existing program
- Created shell scripts that automatically reflect updates in main database to testing database every night

Mentor at Project Tech Conference Hackathon February 2022

- Participated in 3 days hackathon held by Project Tech Conference as a mentor. Duties are helping hackers with debugging their codes, making suggestions based on the team's pace and goals, and sharing ideas

Projects

GuardNav | https://github.com/min2028/GuardNav| JavaScript, React, Redux May-Aug 2023

- Used google auto complete api to allow user to save their favorite location, and implemented a shortcut logic
- Created Figma, hosted short meeting to unblock team members, and suggested changed from end-user perspective

To-do List | https://github.com/Ryo-Ura/Wantedly-Todos| JavaScript, React April 2022

- Used React framework to avoid page reload as keeping tasks always visible would be more useful and convenient
- Worked on front-end and used useMemo, useCallback and useEffect to minimize page refresh and unnecessary update in parent class when creating, editing tasks, and retrieving data from popup form
- Implemented drag and drop functionality is inspired by the Google To-do, and done by using beautifulDnD library

Combat food (nwHacks2022) | https://github.com/Ryo-Ura/combat-food| Flutter, Dart, TypeScript January 2022

- Worked as a team of 4, developed using Simulator, **Received Best UX/UI prize** from Kabam
- Assigned tasks based on teammates' interests, created workflow chart, and added specification note using Miro whiteboard based on the ideas and advises obtained from meeting, and worked on creating pages and transition
- Restaurants create posts by setting price, location, food's images like Yelp, and food type customers swipe images to like/dislike food like Tinder, and store liked item in back-end for checkout
- Implemented page transition to navigate user with minimum page refresh inspired by Amazon, which decreased total number of page user travels to get to purchase screen from 4 transitions to 2 transitions
- When designing, think as the end-user and fix design to prevent users from making purchase decisions at purchase page or accessing purchase page for the same item simultaneously as this makes some users not being able to buy an item

Converting MP4 to ASCII (Hackcamp2021) https://github.com/Ryo-Ura/HackCamp2021, Python, JavaScript Nov 2022

- Created a web application that takes MP4 and converts it into ASCII art as a team of 2 using GitHub and Docker
- Role includes implementation of MP4 to ASCII represented image, which is done by extracting every frame from MP4 and replace each pixel by ASCII code based on pixel's color contrast
- Used the test-driven development, find a bug that skewed the output image and fixed the bug by specifying font that has a fixed width with all characters because "." and letters like "@" have a different width
- Since the conversion of 30 seconds video took about 10 seconds, used multiprocessing for the pixel replacement work which resulted in 80% decrease in the conversion time