Web Application for Robotics Education

Zachery Wiles | Ryan Lunt | Sean Griffith | Herman Hira

Department of Computer Science & Engineering | University of Nevada, Reno | Instructors: David Feil-Seifer and Devrin Lee | Advisors: Dr. Rui Wu, Ben Gallagher

CSE

Abstract

WARE is a full stack web application designed specifically for higher education robotics classes. Incorporating the open-source OpenAI Gym framework, WARE provides a user interface to the robotics environments provided by the framework.

The purpose of the project is to give instructors and students of robotics classes access to the Gym environments using any web-capable device. WARE incorporates the following major features: front-end code editor with syntax highlighting, back-end code compilation, the ability to create user accounts for students and instructors, and an optimized user interface for mobile devices.

Goals

- Create a web-based user interface for the OpenAI Gym robotics environments
- Solve the problem of slow code compilation on host machines that have limited system resources
- Give instructors of robotics courses a tool for teaching students about reinforcement learning algorithms when applied to robots
- Design a learning platform optimized for smartphones and tablets

Key Features

- Users of WARE may sign up as either an instructor or a student to review their students progress or experiment with various OpenAI Gym robotics environments (Figure 1).
- Students are able to experiment with robotics environments assigned to them through a web interface (Figure 3), to allow for use of OpenAI Gym environments with low power and mobile devices (Figures 1-5).
- Instructors are able to create classes for their students to join and assign environments for them to experiment with (Figure 2).
- Instructors are able to view their students' progress in assigned environments through an instructor portal (Figure 6).
- Users are able to view video and textual output depicting the state of the environment throughout the execution of their code (Figure 5).
- Users are able to develop their solutions to an environment within the application or upload their code from their local device (Figure 4).
- User code utilizes python syntax highlighting to ensure a more efficient development experience (Figure 4).

Architecture

WARE was built using Flask and Bootstrap frameworks. WARE is primarily created with HTML / CSS and JavaScript for the front-end, and Python for the back-end. It's architecture is comprised of three subsystems - a front-end, back-end, and a database. The front end consists of all visual user interfaces and individual web pages including the environment page's code box and output windows. The back-end is where security, code processing, and role checking takes place. Lastly, the database is involved in the account management system, managing environment information, and is used for managing the class system.

Future Implementation

- Future implementation of this project is to be able to place this on the university network to allow for students to access and utilize this tool.
- Along with placing this project into a network focus will also shift to making sure security is implemented so that students and instructors information will not be compromised while trying to utilize this tool.
- Along with this new focus new environments shall be added for more variety for professors and students alike to play around with.

Conclusions

Education with regards to robotics is an important field that not many people have an opportunity to dive in to and explore. With a lack of accessibility to robotics education and a generally high barrier of entry compared to other various fields - WARE's goal is to bridge the gap and bring a further outreach to students.

Students will be able to create an account to learn and hone their abilities with regards to completing objectives in the many robotics environments made available to them.

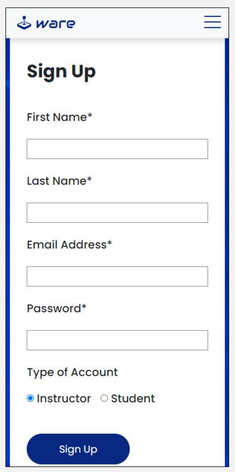

Figure 1: Signup page. A user may sign up As either an instructor or a student. (Mobile View)

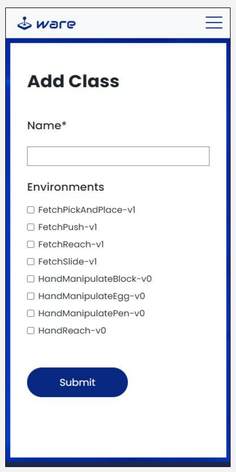

Figure 2: Class creation page. Instructors can create classes for their students to join. (Mobile View)

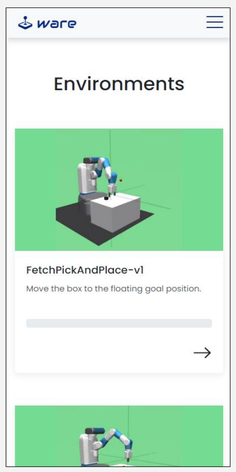

Figure 3: Environment list page. Students are able to view the list of environments assigned to them by their instructor. (Mobile View)

Figure 4: Environment Page. Users may input their solution to an environment directly or through a file upload. User code has syntax highlighting. (Mobile View)

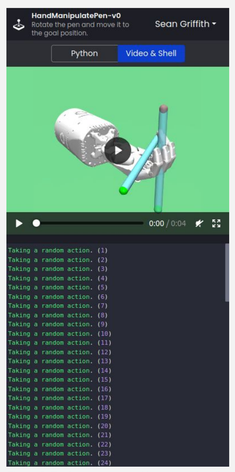

Figure 5: Environment page. Users can visualize the result of their submission through a video playback and textual output. (Mobile View)

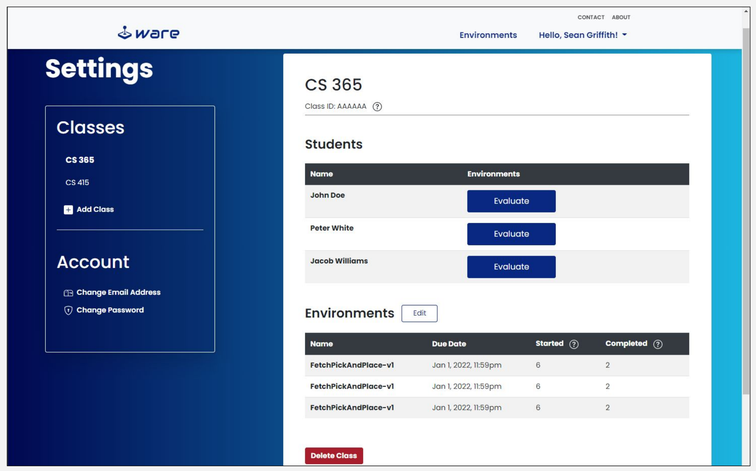

Figure 6: Instructor portal page. Instructors may select a created class, evaluate their students' progress, and edit class information. (Desktop View)

This project was developed in Spring 2021 as part of the course CS 426 Senior Projects in Computer Science