{coding&&community}

Open Source Curriculum Project

Description:

Coding&&Community faces a series of unique challenges in delivering computer science education to middle school and high school students. The club is only able to teach its students for a couple hours every Saturday, and students are generally unable to access course materials from home. We would like to introduce our advanced students to more complex development frameworks such as Flask, or React, but we simply do not have enough class time to build these larger projects with students from scratch.

The goal of our software is to create a system for learning theses extensive frameworks in “fast-forward mode”. The curriculums we design will ask students to implement only the conceptually interesting and educationally valuable parts of a project. To fast-forward through the rest (anything boring, tedious, or repetitive), we will apply modular pre-built Git commits combined with new instructions to move their project to the next stage. These pre-built commits will give the students the starter code they need to complete the next task. The ultimate goal is to create very modular curriculums that allow students to “choose their own adventure”, giving them control over what paths to take and which features to implement. This will result in unique and custom learning experiences for our students.

We understand that many of the frameworks and languages we hope to teach are not taught in any class at RPI. There was so much interest from people who had no prior knowledge that we’ve decided to add a teaching component to our RCOS project. Beginners will first go through our HTML/JavaScript/CSS curriculum, critique it, and help us modify or add anything we missed. Then, they will go off and learn a framework such as React, Node, or Flask to become our resident expert. They will be ready to lead the development of new curriculums for these frameworks. This will allow us to collect a lot of data on what tutorials/exercises worked and what didn’t. By doing this, we will be empowering two generations; the RPI students who are gaining valuable tools used heavily in the real world, and our STEP students who will receive a tailored, well critiqued curriculum.

We are still defining the entirety of our project, but our ultimate goal is to create software that fits a real need of Coding&&Community. We would like to measure the success of this project not by how well it conforms to this proposal, but rather by asking if it solves a challenge faced by our organization.

Team (as of now):

Milena Gonzalez*	Ethan Graf*	Bradford Stone	Clare Cuthell
Corey Boornazian	Edwin Wallis	Gretchen Forbush	Jake Billings
Mack Qian	Manusri Viswanathan	Mitchell Falkow	Peter Shin
Pragati Pant	Randy Wang	Ritvik Vaish	Saketh Dargula
Victor Chan	Zaire Wilson	Jeffrey Chai	

*Project Leads

Timeline:
<u>September 21st:</u> Split group into curriculum and development teams. Research and design software implementation. Identify the best frameworks to turn into curriculums.
<u>October 5th:</u> Create an extremely simple linear test curriculum. Code that will allow students to fast forward through the prototype curriculum.
<u>October 19th:</u> Establish guidelines for how curriculums are designed to minimize conflicts and maximize modularity. Make more complex curriculums. Implement error checking and testing to make sure student code doesn't interfere with the rest of their project.
<u>November 2nd:</u> Begin design of the MVP (minimum-viable-product) curriculum. Implement project branching.
<u>November 16th:</u> Thorough development of the MVP curriculum. Implement a system for delivering instructions in addition to the pre-built commits.
<u>November 30th:</u> Anticipate the unanticipated. Wrap up loose ends.
<u>December 14th:</u> Build a basic website to host the curriculums and make them available as open source content.