Educational Programming Through Interactive Media

Requirements Backlog - Team 2

Anant Goel, Don Phan, Jason Rahman, Jon Egeland, Josh Selbo, Levi Linville

Problem

In current society, there is a high demand for Science, Technology, Engineering and Mathematics (STEM) specialists, especially Computer Scientists. To meet this need, there is currently a demand in the education world for tools to teach basic programming concepts and logical reasoning in a way that engages students, maintains their interest, and provides positive reinforcement.

Background

Audience

Currently, there are still many schools around the state of Indiana that do not offer computer science related courses. Our project can aid Phil Sands in his outreach program to help spread information to middle and high school students about computer science as a subject.

Similar Applications

There are a few existing web-based tools that are similar to what we are trying to create. For example, code.org is a site that teaches students how to program through games. These sites attempt to find a balance between giving the students a taste of thinking like a computer scientist and working with the lower level details of computer science.

What we want to accomplish is to meet in the middle, melding a seamless mixture of thinking like a computer scientist and programming; to find harmony between the boilerplate and sense of discovery. We do not want to simply focus on individual topics. Instead, we want our game to tie all those skills together in a polished package.

Existing Limitations

The learning tools that currently exist tend to focus on teaching either the critical thinking skills of a Computer Scientist or actual programming skills (languages, data types, etc.). Our application attempts to bridge the gap by introducing basic programming skills and then using those skills to help teach critical thinking.

Additionally, existing tools generally lack a continuous storyline, which detracts from the user's immersion in the game. Our application will be based on a single storyline which progresses with the user, allowing them to continue the story by completing levels.

Requirements

Functional

1. As an administrator, I want
    a. to manage users
    b. to create users
    c. to delete users
    d. to reset a user's password
    e. to reset a user's progress
2. As an instructor, I want
    a. to manage classes
    b. to create classes
    c. to edit class preferences
    d. to add students to classes
    e. to drop students from classes
    f. to set a class's language (time permitting)
3. As an instructor, I want
    a. to view my students' progress
    b. to view my students' achievements
    c. to view my students' submissions
4. As a student, I want
    a. to manage my account
    b. to register for the site
    c. to manage my information (name, email, etc.)
    d. to reset my password if I forget it
    e. to enroll in classes
5. As a student, I want
    a. to interact with the game
    b. to choose and name my character
    c. to move my character around the map
    d. my progress to be saved and restored when I return to the game
6. As a student, I want

   a. to play a level
   b. to read tutorials to help me with the gameplay
   c. to have hints available for tough problems
   d. feedback about my syntax and coding mistakes
   e. to see a demo of my executed code before I submit it as a move (time permitting)
   f. to choose my programming language (time permitting)
7. As a student, I want to play a challenge
8. As a student, I want to view my progress

Non-Functional

Architecture

The application should use a common client-server paradigm to provide separation of concerns between the client and server. This will facilitate easier division of effort between the development team members and allow the UI and backend component to evolve in a more flexible manner.

For ease of development, we want to develop the application using Ruby on Rails for the majority of backend functionality, and JavaScript, HTML5, and CSS for the client side functionality.

Performance

To ensure a seamless and fluid experience for student playing through levels and challenges, the server response time should be less than 500ms and ideally less than 250ms. On the client-side, Javascript execution should be less than 10s (with a variable timeout). The overall goal for performance is to achieve a responsive environment for the user.

Security

Security is not a crucial feature for this application. The consequences of security breaches are relatively low, since no sensitive information such as banking or medical records are stored. Account passwords will be encrypted using standard password-encryption techniques, potentially we could employ OpenID for authentication as well. Interaction between clients and the server shall be encrypted using SSL/TLS to avoid basic session hijacking attacks. Taken as a whole, we adopt the philosophy of utilizing simple, easy to implement security measures that provide a high benefit to cost ratio. A strongly secure system is not required and is not worth the effort needed.

Hosting and Development

The application will be developed and deployed using virtual machines, with different virtual machines fulfilling different roles (database, Ruby on Rails backend, remote code execution/grading, etc). Using virtual machines with well defined roles for each VM, the application will be easier to scale out to a larger number of users if needed. For example, as the user count increases, more machines can be added to run student code input and perform grading.

Appearance

The application should appeal to the target audience in a positive and engaging way. Design will be gender neutral to ensure the game appeals to a broad an audience as possible to encourage a more diverse base of computer programmers in the future. Music and design are important to keep students engaged and for the success of the application.

Grading Isolation

We need to run the code in a restricted sandbox to avoid attacks by malicious users and poor quality code. This is key to ensure a good user experience for the students, instructors, and administrators of the application.