PyAccountant: Redesigning Programming Education for Accountants

Albert Tung
Ty Feng
Ken Chiu
Jingxian Wu
Chi Zhang
UC Berkeley
Berkeley, CA, USA
{albertytung,tyfeng,kenchiu,jwu5,cz}@berkeley.edu

ABSTRACT

PyAccountant is a website designed for accountants to learn programming through an engaging and interactive story involving an accountant and demons. It is intended to facilitate learning the basics of programming in Python and also kindling an interest in programming by showing accountants the various ways in which Python can be used. This will ideally allow accountants to automate their programming tasks and reduce dependence on close-source and expensive accounting software.

CCS CONCEPTS

• **Social and professional topics** → **Computer science education**; *Adult education*; *Automation*;
• **Human-centered computing** → *User studies*; *Heuristic evaluations*; *Walkthrough evaluations*;

KEYWORDS

Computer science education; automation; human centered design.

CHI'19, May 2019, Glasgow, UK

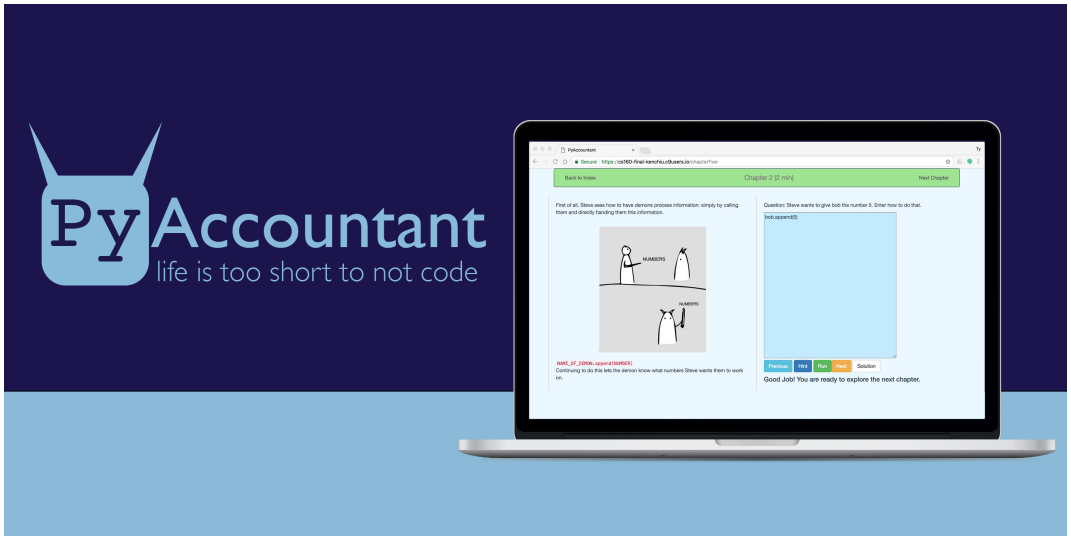

Figure 1: PyAccountant Project Image.

ACM Reference Format:
Albert Tung, Ty Feng, Ken Chiu, Jingxian Wu, Chi Zhang. 2019. PyAccountant: Redesigning Programming Education for Accountants. In *Proceedings of ACM SIGCHI conference (CHI'19).* ACM, New York, NY, USA, 15 pages. https://doi.org/10.475/123_4

INTRODUCTION

Motivation

Accounting majors in university only learn the most basic of programming, if any, sometimes even foregoing learning even Python and only focusing on SQL for database management. Clearly, there is a need to teach accountants programming and its applications. This is even more relevant for those accountants who graduated decades earlier. Although accountants could use other online programming learning applications or textbooks for learning programming, none currently focus on accounting and its uses. Another issue is those other applications are usually fairly dry and may not be particularly engaging for beginners. Our application is situated in this gap, combining interactivity with vivid and descriptive analogies in order to facilitate learning. Our narrow focus ensures that we can capture interested in this unexplored niche of accountants attempting to learn programming.

Overview of Design Process

Brainstorming. Our group spent time finding new and novels ideas and an unexplored market.

Hierarchical Task Analysis. We analyzed how people learned programming.

User Research. We interviewed people on a programming/learning task.

Synthesis. We combined the users' critique of existing methods and thought of ways to improve their experience.

Personas. We created an "ideal" user and used that in order to inform out design process.

Initial Interaction Sketches. We created sketches of what our finished product should look like.

Wireframe. We created a wireframe based on feedback of our initial sketches.

Mood Board and Motifs. We conducted a formal analysis of images with the theme of programming and accounting. We created a color palette to guide our design process.

Peer Critique. We asked peers to criticize our design and on improvements to be made.

Product Redesign. After negative reviews of our design, we overhauled it.

Medium Fidelity Prototype. We created an ideal of what our final product should look like in Adobe XD.

Heuristic Evaluation of Mid-Fi Prototype. We rated our mid-fi prototype and determined any usability issues that might arise, and rated them based on severity.

Final Logo. We created a logo to use on our application for marketing purposes.

Project Image. We created a project image to represent our application.

Implementation. Using HTML, CSS, Django, Bootstrap and Javascript, we implemented our design.

Final User Evaluation. We had users evaluate our final design.

Synthesis. We took user feedback and used it to inform possible areas of improvement for the future.

Solution and User Evaluation Results

Our solution was to make a programming website with user interactivity and an interesting and engaging story. Our users stated that, while our idea was well-thought out, there were some implementation details that they had issue with, such as notifications that were not eye-catching or text that did not seem well-formatted.

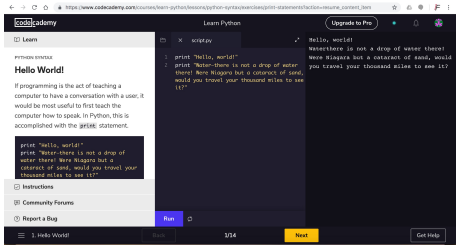

Figure 2: Codecademy, an online coding tutorial website for a generic audience of programmers

EXISTING APPLICATIONS AND DIFFERENCES

Currently existing many websites that offer coding tutorials or courses, including Codecademy, CodeAnywhere, KhanAcademy, and various MOOCS (Massive Online Open Courses). But our project is different from each of them. These each have drawbacks concerning efficiency and relevance that we are able to address via our project.

Codecademy. Codecademy is an online coding tutorial website.

CodeAnywhere. It has the stringent requirement that users must have a fast and persistent Internet connection. Having a slow Internet connection makes interaction difficult (page loading time), while having an inconsistent connection renders the service unusable (users are unable to even edit code). Our application also features a narrower focus, which means that it does not impinge on the niche of CodeAnywhere.

While MOOCS currently exist which address learning how to program and feature a large, interactive community and reasonable amounts of coursework, this may not always be suitable for a business professional. Business trips can hinder a professional's ability to work on large programming projects and readings. These may eventually force a given user to quit to due work-related reasons. Another issue that also exists with CodeAnywhere is the lack of emphasis on procedural programming. This results in accountants learning vast amounts of information that they might not necessarily need. Ours offers quick, instant feedback in an application directly related to accounting.

BRAINSTORMING

Our group initially started by creating a list of fifty ideas. Following this, we discussed which idea had the strongest user need and fulfilled project requirements. We narrowed our options down to seven ideas and continued questioning need until we decided on a coding interview app. In order to narrow focus we decided to cater to accountants who needed to learn how to code. Initially, our ideas was merely creating a story about an accountant, believing using him to relate programming concepts to the real world would draw accountants to using out website.

HIERARCHICAL TASK ANALYSIS

Our group first created a hierarchical task analysis on how people would learn coding so that we could focus on specific components in which to focus on. We ultimately decided that the information that people had to parse could be made more interesting (Step 1), and that the feedback that people receive could potentially be made more useful/entertaining (Step 4.1).

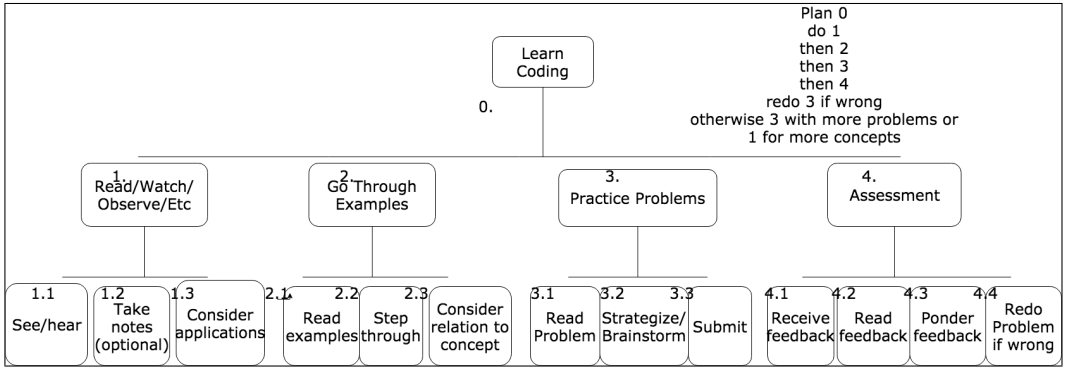

Figure 3: Hierarchical Task Analysis

USER RESEARCH

In order to create a coding website, we first conducted user research. We did so by gauging the usage of Codecademy. To decide target users and finding user needs, we contacted 5 persons from five different fields (with one participant being unable to finish interacting with Codecademy due to time constraints), observed their thought process and interviewed them on their feeling of their experience.

Synthesis

From our interviews, we realized that there were common complaints about Codecademy: Users were unable to interact with Codecademy due to the need for a persistent connection. Lack of user engagement due to uninteresting instructions.

In order to solve this issue, we wanted to have 'Steve' (our character as an accountant in the story) to be progressing from Junior Accountant to Senior accountant, which would be illustrated on the title page. Similarly, our example would have a direct relation to accounting, by relating each concept as a method of processing spreadsheet data more efficiently.

Personas

We picked one of our interviewee who was a tax accountant as the person we are designing for and created a persona for him. This helps us to learn more about the personal needs and expectations of tax accountants.

Tax accountant. A middle-aged tax accountant who lives in the Bay Area. He currently spends most of his day inputting data for customers into TurboTax (especially during tax season). He does not see much opportunity for advancement, but thinks that, with proper programming skills (to automate tasks), he could spend much less time on simply inputting data and spend that time on personal and professional development (networking, conferences, meeting with friends). He doesn't know where to start. The last time that he looked at programming, he just saw something about 'Foobars?' It seemed pretty impractical.

INITIAL INTERACTION SKETCHES

Our group began with all of us each creating an initial interaction sketch. This was followed by brainstorming and discussing which features in our wireframes were considered to be the most useful. We used those ideas in order to inform our wireframe and to have a basic idea of how to implement our website.

WIREFRAME

We refined our initial sketches and combined our ideas into this wireframe.

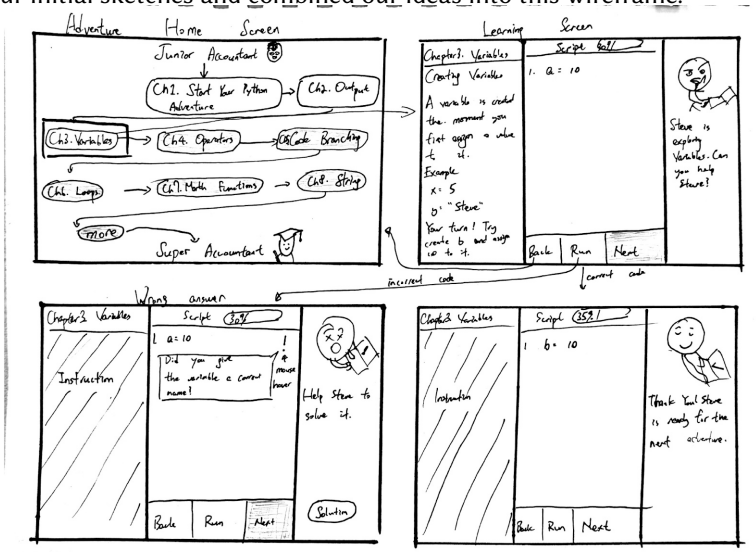

Figure 4: Moodboard

MOODBOARD AND MOTIFS

After conducting a formal analysis on images related to the theme of programming and accounting, we created a mood board (Figure 3) and three motifs (Table 1) for the coding interface. We used the mood board to inform us in our designs decisions.

PEER CRITIQUE

In order to gauge the effectiveness of our website design, we received three peer critiques in order to determine how well-made our user interface was. Common complaints were that 'Steve' did not seem particularly useful or well-made. In fact, one user even wanted to remove Steve entirely. Our group continued to brainstorm for ways to make Steve more relevant, as having him was necessary to fulfill project requirements.

PRODUCT REDESIGN

After another period of brainstorming, our group drastically changed the story for user interest. Instead of simply having 'Steve' (our accountant in the story) simply learning how to program, we instead created an analogy in which he interacted with 'demons' in order to have them do his bidding for him. In creating this analogy, we managed to create a project with novel features and were able

Table 1: Motifs

1. Sketchy Interface
In order to have users relate to the story, the illustrations will mostly be hand-drawn so that characters are more personable. Ideally, this will also capture user interest, as the format will be more similar to a picture book than blocks of text.
2. Bright Colors
This motif contrasts with Motif #1: it will have a bright background color. The keywords colors are warm colors – except for red – like deep yellow, orange. Study guideline colors are the same as they in motif 1. If proper colors are chosen, all color can be highly saturated for better contrast.
3. Simplistic Interface
Common complaints while using Codecademy were that there were too many buttons and features that confused first-time users. Therefore, our interface aims to be as simple as uncluttered without removing vital features. Ideally, this will allow users to focus more on the programming task at hand than they would otherwise.

Figure 5: Final Logo of PyAccountant. Implemented using Adobe Illustrator. The logo uses monospace to emphasize the coding aspect, while the horns indicate the demonic theme.

to increase user engagement (as evidenced by instructor feedback). Another indirect advantage of having an engaging story is that it will help users encode information, due to its novel and memorable nature. After receiving instructor feedback, we finally solidified our design and only needed to work on our implementation details and to solidify our website design. Fortunately, this only required us to change parts of the story, leaving the interface untouched. This only necessitated redrawing the user interface screens and completely rewriting the story. The user interface screens were done in Adobe XD, the logo was done in Adobe Illustrator, and the story was created using Google docs, while the drawings were done on paper.

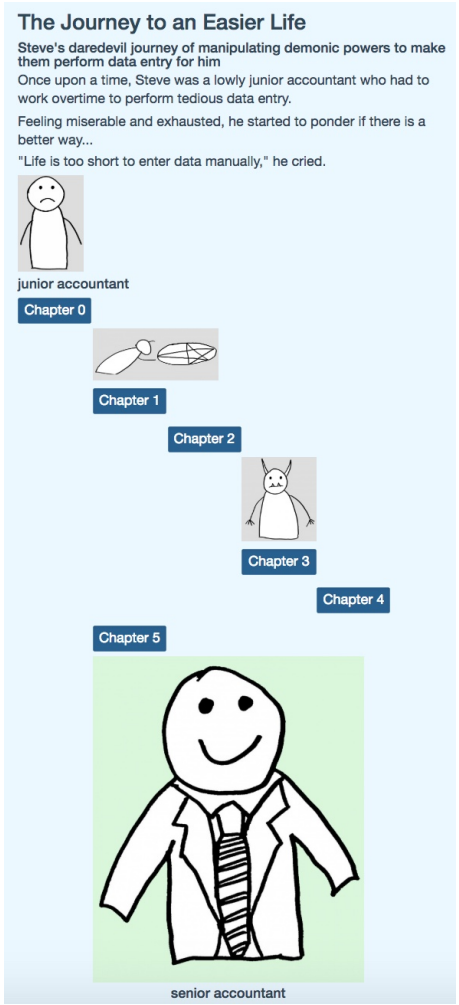

Figure 6: Index page; Allows users to navigate through chapters

MEDIUM FIDELITY PROTOTYPE

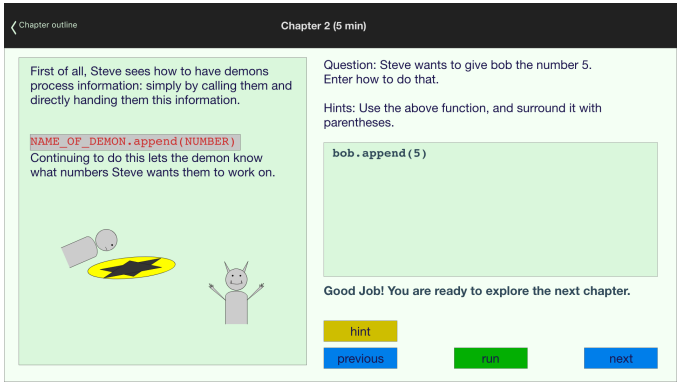

Figure Our team's mockup of the website, implemented using Adobe XD.

HEURISTIC EVALUATION (MEDIUM FIDELITY PROTOTYPE)

We conducted a heuristic evaluation according to Jakob Nielsen's paper "Heuristic Evaluation of User Interfaces", and identified six key issues that violate those heuristics. We gave a severity rating, numbers assigned to each issue indicating its severity according to the following:
0 = I don't agree that this is a usability problem at all
1 = Cosmetic problem only: need not be fixed unless extra time is available on project
2 = Minor usability problem: fixing this should be given low priority
3 = Major usability problem: important to fix, so should be given high priority
4 = Usability catastrophe: imperative to fix this before product can be released
We considered frequency, impact, and persistence as the primary reasons for giving those severity ratings. The heuristic evaluation table can be found in Table 2.

IMPLEMENTATION

The technologies that our group used, aside from basic web technologies, were JavaScript and Bootstrap. Our server was hosted on Cloud9 and was run with Django. JavaScript was used in order to check the user's answers and to provide hints when necessary. The choice of JavaScript was informed by the need to keep a consistent page and to provide the maximum amount of interaction. Bootstrap provided the necessary technology to smoothly and seamlessly separate the columns, while providing built-in styling in order to maximize productivity and style. In order to find our color palette, we considered the color themes in our mood board and picked dark blue and green as our primary colors.

Table 2: Heuristic Evaluation

#	Issue and evidence	Broad Heuristic	H #	Severity Rating
1	Users have no way of knowing whether or not they submitted something properly or if their input is busy processing. For example, during form submissions, there are no loading icons or icons to indicate whether or not something is loading.	Visibility of System Status	1	1
2	User can potentially run into an infinite loop, when submitting code, and would be unable to progress, with no indication of this occurring.	Visibility of System Status	1	4
3	Users do not have any indication when there are syntactical errors in their written code.	Error Prevention	5	2
4	No keyboard shortcuts/accelerators for submitting code or browsing.	Flexibility and Efficiency of Use	7	1
5	When code submitted is incorrect, there is no indication where the error is or how to fix it.	Help Users Recognize, Diagnose, and Recover from Errors	9	4
6	No current documentation for application. Users are expected to understand the website without any instruction.	Help and Documentation	10	2

For our index page, in order for users to select a chapter, we implemented an organized grid via Bootstrap.

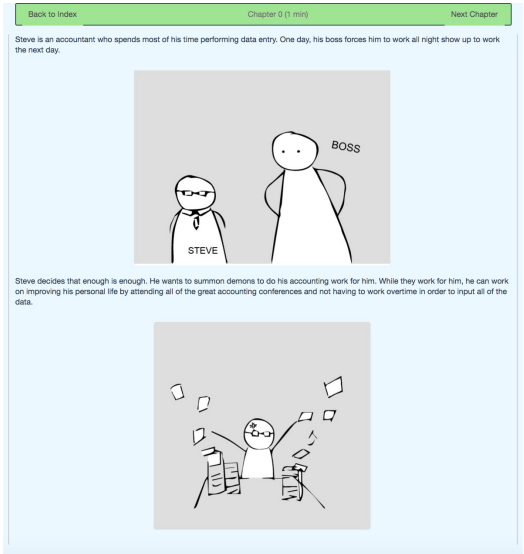

Figure 7: The initial chapter, provides overview of story

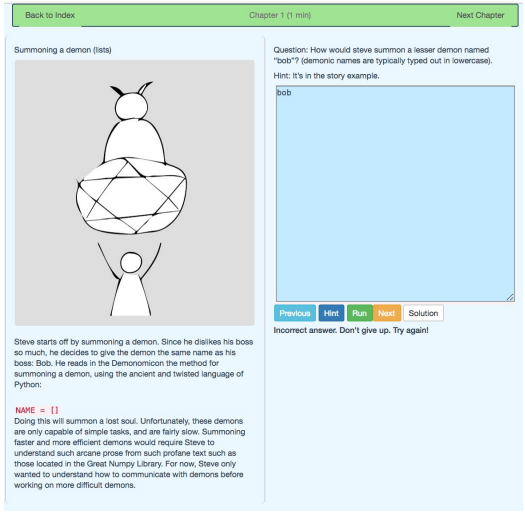

Figure 8: Users are notified when their solutions are incorrect.

FINAL USER EVALUATION

We started out by finding users. We found our target user, an accountant from the Bay Area, to use our interface. Following that, we took our user to a secluded room, free from distractions and used a think-aloud. Following this, we asked a series of questions to evaluate our design decisions. We used the remarks and answers to questions in order to fix any perceived issues in our interface.

Method

Research question: Is our UI easy to use and interesting? Is it enticing enough to learn coding as a beginner?

User recruitment:

We asked five students, who have no prior experience of programming to be our users. This was done in Moffitt and MLK Student Union building.

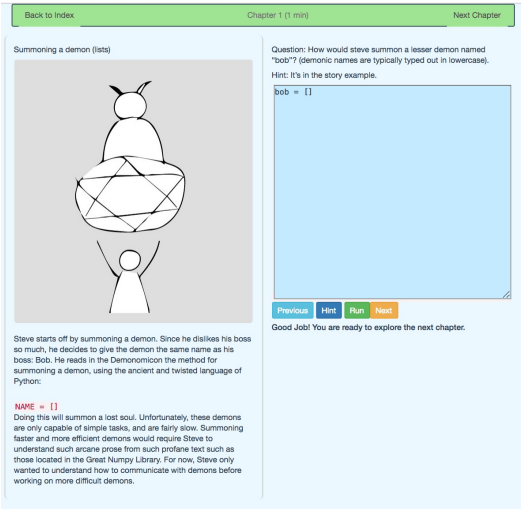

Figure 9: Users are told when their solutions are correct

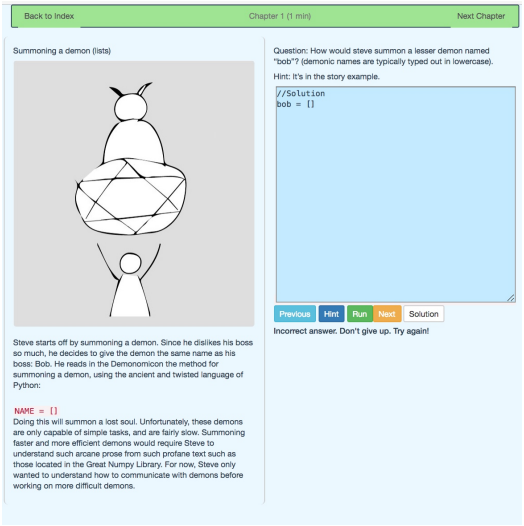

Figure 10: Users can be given hints and the solution if they get the answer wrong

Procedure:

- Greet the user (1 min)
- Introduce ourselves, and summarizing/contextualizing our design solution (2 min)
- Give a general outline of what the user will be testing (navigation around the website, learn the python, do they find learning python is interesting) (1 min)
- Think aloud (user using the interface) (10 min)
- Ask questions for what they feel about this UI after they finish (1 min)

Result

Things that users like:

- Images are cute
- Good choice of color
- Hints are helpful
- The story makes coding more fun and more motivating
- It has all five chapters

Figure 11: Congratulations; When users are finished with all of the coding chapters, they are greeted with a congratulations page and are invited to share their accomplishment with the world via Linkedin, Facebook, etc.

- Solution is helpful

Things that need to be improved:

- Some images are too big
- Some texts are too small
- The story could confuse users about learning Python
- They want a real console
- Terminologies are not clear

Click here to see final user evaluation script.

Synthesis

During our final user evaluation, users had a mostly mixed reception to our interface.

Need more work on visual hierarchy

Although our concept was solid, graphical design choices tended to detract from users' opinion of our product. A common complaint was that the text size was too small and there was not enough visual hierarchy in terms of text for users to understand what was important and what was not. For example, our "hints" simply appeared on the screen, with no notification that they had appeared other than "Hints:" preceding any hints that were displayed.

Make story length consistent in every chapter

Our story was not of consistent length, which led users to be annoyed and to skip text when it came to our longest chapter (chapter 3). These led to users skimming through text and glossing over important information, while also confusing users when they did not know where to find the hint.

Make sure terminologies are clear

Some design choices were also a matter of taste, with some users reporting positively, and others reporting negatively. Our analogy of "demons doing accounting for you" led to some nomenclature that could not be cross-referenced; for example, in order to instantiate a list, we 'summoned' a 'demon,' and in order to access list elements by indexing, we used 'demonic' counting. Unfortunately, this simply a tradeoff between using a novel analogy and simply instructing users using programmatic terms.

Keep the good elements

Positive aspects of our design mostly involve elements that we chose to include. Our pictures were considered visually attractive to most of our users, and contributed to interest towards the story. Our hints were also consistently helpful (when users were able to find them), and minor implementation details such as string matching and stylizing code in the story helped users to understand the material better. Overall, our design was grounded in solid principles, but difficulties in implementation led to some suboptimal design choices.

CONCLUSION

Overall, our project was deeply informative of the design process in a collaborative, real-world environment. First of all, we initially had difficulty in finding an idea that would be able to fulfill the project requirements (especially in finding a niche that was both unexplored and fulfilled a user need). Our team performed well in allocating tasks fairly so that each team member contributed similar amounts of work, especially in areas that they already had experience in. Difficulties in our design process were mostly technical. The most difficult hurdles were contending with software-as-a-service

websites for coding our website and agreeing on which direction to take our app. Although we had different ideas on which direction to go for project at multiple steps in the design process, it was necessary for our project manager to decisively select a particular direction in order for us to progress. Similarly, our team did not have much experience in implementing a website – given our current level of experience, there were some design elements that we agreed could have been improved, but we were unable to modify satisfactorily. Overall, this has proven to be a rewarding and instructive experience that has provided invaluable experience for the future.

LINKS

Source Code Repository

Github Link

Presentation Slides

Google Slides

Medium Article

Medium Article

Concept Video

Concept Video on Youtube

Walkthrough Video

Walkthrough Video on Youtube

High-fidelity Implementation Prototype

Our live implementation prototype demo hosted on Firebase

Poster

Final showcase poster (pdf)

Chapter Stories

All Stories on Google Doc