WHITEBOARD

Team 18 - White Board Product Backlog

Chunao Liu, Anurag Shah, Jenna Zhang, Yierpan Abuduwaili, Michelle He, Jingyuan Yang

Problem Statement:

Modern Coding usually requires a laptop with an IDE or a compiler, but there are many situations where someone must test a code sample without that code even in a textual format. For example, an interviewee preparing a coding interview will have to write their code on a piece of paper or on a whiteboard, and without a compiler, one can only debug by observing their written code or typing it out in full, a waste of time for all parties involved. Our project aims to develop a novel “IDE for Written Code” that can support typeform code or scanning hand-written code, compile the code and return the terminal output and stack trace back to the user.

Background Info:

Definitions:

- Typeform code: a code sample that is typed, but not in a computerized text format. Examples are code printed onto paper, or a screenshot of code, but do not include code written by hand.

Audience:

Code, as a textual format, has never been restricted to a computer interface; as such, it is common to find code typed on mobile apps, taken as a screenshot image, or even written down on paper. On those occasions, compiling and running code is impractical. It is an unnecessary hassle to type the code into an IDE or editor on PC, and compile it again in order to see if it runs correctly. For people in such a situation, an app that could easily scan and compile the code, would save minutes of effort that can go into analysis.

Similar Services:

There are already numerous Integrated Development Environments (IDEs) on the market that allow developers to build software, apps, or services, in a variety of languages, such as Visual Studio, IntelliJ IDEA, Xcode, Eclipse, PyCharm, etc. They all allow users to compile and debug code, but most of them provide programming capabilities through a text editor or a GUI. No IDE, or editor of any sort, allows you to scan in code in a non-textual representation. There are also many ios/android apps for code compiling and debugging, such as Pico Compiler, JSAnywhere, and Python2IDE, but they similarly lack the functionality of reading hand-written code. Finally, OCR as a tool has existed for many years, and is constantly improving in its accuracy, even with handwritten text, but it is rare for an OCR service to support the full ASCII set like most programming languages require, and no OCR service has such an integration with a compiler.

Limitations:

While there are many existing Apps using OCR as a data input device to read numeric and alphanumeric characters from printed or handwritten documents, the accuracy of scanning characters depends solely on the clarity of writing and how neat the handwriting is. Existing services for handwritten OCR are limited to specific character sets, though a personally trained model will somewhat alleviate these concerns with the production of character data. Further, OCR deals poorly with whitespaces and space width, which limits the languages such a service can support. Many of the other Apps have poorly designed user interfaces and do not provide any feedback on the scanned document. The images produced will need a large, dedicated database for storage. Our goal is to address these problems by creating an App that scans well-structured programming languages like Java, C/C++, C#, Perl, among many others. that do not require spacing as a syntactical form. Further, we will provide feedback to the user, especially in the case of compilation errors, and store the image for later use.

User Story / Functional Requirements:

As a user, I would like to have a welcome page when I open the app.

- As a user, I would like to have a "remember me" feature at login.

- As a user, I do not need to log in again if I’ve logged in before and did not log out.
- As a developer, I would like to send a password resetting link to users via their email address when they forget their passwords.
- As a user, I would like to be able to create my own account.
- As a developer, I would like to link users’ accounts with their emails/phone numbers.
- As a user, I would like to have a profile page where I can edit/update my info.
- As a user, I would like to be able to change the theme of the app (light/dark).
- As a developer, I would like to have a navigation bar which directs users to different pages.
- As a developer, I would like to prompt users to re-login when the account is inactive for a week.
- As a user, I would like to be able to deactivate/reactivate my account.
- As a developer, I would like to delete a deactivated account permanently if the user does not reactivate within 90 days.
- As a user, I would like to use the Apps feature without creating an account.
- As a user, I would like to have a way to reset my password.
- As a developer, I would like to load pictures and snapshots faster using multi-thread.
- As a user, I would like to have a profile picture and a banner.
- As a user, I would like to see if there is any syntax error in my code, have the lines physically underlined with a marker on my code image, and have a popup after clicking on that inform me about the full error details encountered at that line.
- As a user, I would like to access an archive of the scanned documents in digitized text form that can be saved in an editor app (such as the Note app in iOS).
- As a user, I would like to retake a photo of the code if the image taken is not satisfying.
- As a user, I would like to join/create/leave multiple teams.
- As a team member, I would like to see who is the coordinator/creator of the team.
- As a team manager/member, I would like to see the team members.

- As a team manager/member, I would like to be able to access all shared code between the team.
- As a team manager, I would like to remove a user from the team.
- As a user, I don't need to see the login page as I re-enter the app starting from the second time.
- As a team manager, I would like to have access to view and change the team members' code.
- As a user, I would like to be able to upload a photo from my phone's library in place of taking a new photo.
- As a user, I would like to save the OCR-generated code on my device storage.
- As a user, I would like the backend to be able to figure out what language my code is in, and use the correct compiler.
- As a user, I would like to delete the archive of a scanned document if it is no longer needed.
- As a user, I would like to have my C/C++ code compiled & run.
- As a user, I would like to have my Java code compiled & run.
- As a user, I would like to have my C# code compiled & run.
- As a developer, I would like to design various user roles so that users with different roles can access different features.
- As a developer, I would like to allow users to have multiple roles.
- As a developer, I would like to use cacheManager to load data so that the backend does not need to retrieve data from the database every time there's a request so that data access latency is decreased (if possible).
- As a developer, I would like to store each user's authentication information in a SQL query.
- As a developer, I would like to be able to retrieve entries of a specific user's authentication information during the user's login session
- As a developer. I would like to keep track of each user's current status, either logged in or not logged in
- As a developer, I would like to be able to store each user's uploaded pictures in a relational database
- As a developer, I would like to be able to retrieve the picture after the OCR detection and send it back to the front-end
- As a developer, I would like to keep track of each user's team information in a relational SQL query
- As a user, I would like to be able to access photos I have taken previously

- As a developer, I would like the database to be able to store the image filename, detected code, and compilation results/error messages
- As a product manager, I would like to see continuous delivery on the backend.
- As a product manager, I would like to have unit tests for both the front-end and back-end.
- As a user, I would like to be able to process my pictures of both written and typeform code
- As a user, I would like to be able to take photos from slight angles or at a slight tilt, and not have that affect the app
- As a developer, I would like to have a separate classifier to differentiate between written and typeform code
- As a developer, I would like to have an image preprocessing suite that can de-skew and remove lines/spots
- As a developer, I would like to have a separate segmentation algorithm for typeform code images
- As a developer, I would like to have a separate segmentation algorithm for handwritten code images
- As a developer, I would like to have a processing script to convert segmented characters into the required format and grayscale coloring
- As a developer, I would like to have a rich dataset of at least 100 images for each relevant ASCII character, or have a model already trained on a similar set
- As a developer, I would like to have a dedicated HCR model for typeform code images
- As a developer, I would like to have a dedicated HCR model for handwritten code images
- As a developer, I would like to ensure that both models can account for the full ASCII set of characters (where they can be printed, and excluding whitespaces)
- As a developer, I would like to receive an exported ASCII character corresponding to each identified character, and ensure the output is in a format suitable to be compiled
- As a developer, I would like to ensure that each model can correctly place spaces in the program where necessary for the compiler

Non-Functional Requirements:

Performance expectation:

Depending on the actual code to be compiled and the user's internet connection, the response time from the server may vary, but each user should expect to get a response time of at most 10,000ms. We plan to allow 10 users at most to connect and compile their code on our server simultaneously, but this can vary depending on our server.

Security:

We will require all the users to create their own account with their credentials being their email address and password.

Source code files are extremely sensitive and user security is the first priority of WhiteBoard. By using Prepared statements instead of plain dynamic queries, we can prevent SQL injection attacks with a negligible performance impact. We will also limit each user's maximum operations to prevent DDOS attacks or exploiting our server for cryptocurrency farms. By limiting each user's memory space and using the global thread lock, WhiteBoard will ensure that each user will have the same amount of resources so that everyone can properly use our application.

We will restrict users to 1 image processed per 2 seconds. We will restrict images sent to the backend to be no more than 5MB, and the total storage space per user account to 100MB.

Platform compatibility / Deployment:

We will use React Native as our front-end framework, which enables us to perform cross-platform development. Our final product will be able to be deployed in both IOS and Android environments. Our front-end and back-end are very separate, which allows us to concentrate on specific fields without interfering with other's work. It will also be easier for anyone to maintain this app because adjusting the front-end will not interfere with the code integrity of the back-end and vise-versa.