InTouch: CPSC 490 Project Proposal

Student: John Amadeo Daniswara, Computer Science ‘19
Advisor: Kyle Jensen

Goal

I hope to build an Android app that allows friends and relatives of inmates to easily send physical mail to their loved ones in prison.

Background and Motivation

A. InTouch Project is a non-profit that went through the TSAI City (formerly the Yale Entrepreneurial Institute) accelerator last year. Their mission is to use technology to help inmates and their loved ones keep in touch, relying on long-standing research that increased contact between inmates and friends/relatives reduces recidivism.

InTouch recently approached me with the idea of building out an Android app that would allow users to easily type out or dictate (via speech-to-text) letters that would then be sent as physical mail to the inmate’s prison. This app directly addresses our target market’s needs by providing a free and easily accessible alternative to the logistical hassle of sending physical mail and expensive e-messaging solutions by for-profit companies.

B. Moreover, this project will be an opportunity for me to learn and gain hands-on experience implementing various computer science concepts and technologies. These include:

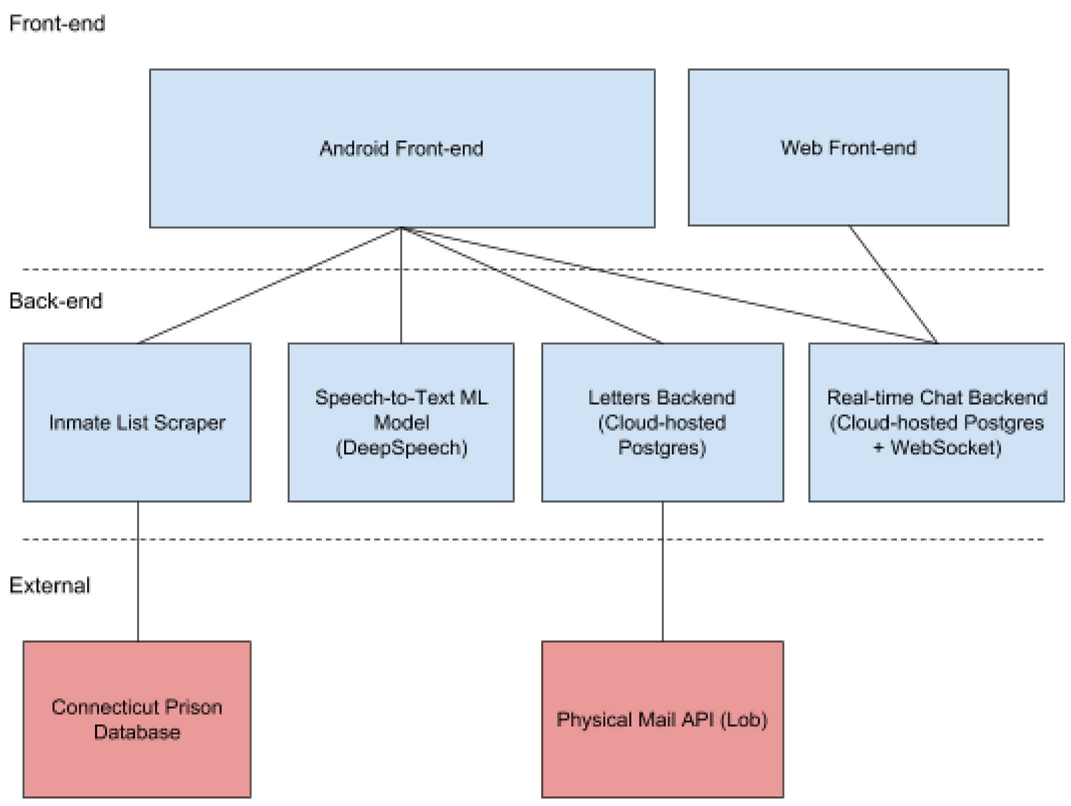
- Learning Android development and mobile UI design
- Learning Go and RESTful API design
- Database schema design and security
- Learning scalable architecture patterns through usage of a cache, reverse proxy, message broker, etc.
- Integrating with 3rd party APIs
- Deploying an ML model in production
- Writing rigorous test suites

(please see the sections below for a complete description of the technical architecture)

These concepts and technologies are commonly utilized in industry and hence highly relevant for my development as an engineer. Designing a scalable and maintainable architecture and codebase will be highly practical when I work on large-scale systems in industry.

Technical Architecture

I present a high-level overview chart and a summary that lists all the components:


Front-end
Android Front-end
Web Front-end
Back-end
Inmate List Scraper
Speech-to-Text ML Model (DeepSpeech)
Letters Backend (Cloud-hosted Postgres)
Real-time Chat Backend (Cloud-hosted Postgres + WebSocket)
External
Connecticut Prison Database
Physical Mail API (Lob)

Front-end (Android)

- UI for writing letters
- UI for using voice dictation to compose letters
- UI for searching the inmate the user wants to send a letter to
- UI for managing list of letters
- Chat UI for communicating with InTouch members for customer support

Front-end (Web)

- Chat UI for InTouch members to respond to customer support messages

Back-end

- HTTP server and reverse proxy (NGINX) for processing API calls from the client
- Cloud-hosted relational database (Postgres)
- API integration with 3rd party mail API (Lob) for converting letter text into physical mail sent out to the inmate
- Deployed open-source ML model (using Mozilla's DeepSpeech) for speech-to-text and server to stream audio from the client
- WebSocket server and message broker (RabbitMQ) to implement real-time chat
- Scraper that periodically retrieves and updates list of all inmates in CT (using Colly)

Languages/Frameworks

The Android front end will be implemented in Java. The web front end will be implemented in Javascript using React. The server backend will be implemented in Go.

Testing

I plan to write test suites with extensive coverage for each technical component, using popular testing frameworks. I also plan to set up a system that automatically creates a new build of the app and runs it against relevant tests whenever a new pull request to my Git repository is made.

Implementation Timeline

Note that testing is incorporated into the development of each component.

Week 3-4:

- Learning Go and Android
- Development of user stories
- Database schema design

Week 5-6:

- Implement Android front-end (with a stub backend server providing test data) for composing, sending, and managing letters
- Setup automatic build/testing system

Week 7-9:

- Implement HTTP server for handling API calls
- Write inmate list scraper
- Integrate backend with the physical mail API

Week 10-12:

- Deploy ML speech-to-text model
- Write server functionality for streaming audio data from client
- Implement Android UI for using speech-to-text

Week 13-14:

- Write WebSocket server and set up message broker to handle realtime chat
- Implement Android/web UI and logic for realtime chat

Deliverables

- Final project report describing app features and the system architecture
- Fully functioning Android app with the above features
- Video demo of the app