Matt Birds LINE Portfolio Read Me

The following are the various pieces of my portfolio, detailed here for easier understanding.

Code

LINE Login

code/lineLogin.php
code/login.js

A simple login using the LINE Login V2.1 API. It is pretty basic, but novel in that it handles the various login interactions from the process and redirects the user back to the page they came from rather than a fixed page assigned in the LINE account manager. On the live webpage, there is also separate login php and js files. Please ask if you want to see.

LINE Bot

code/lineBot.php

The core logic of the my iteration of the LINE Messaging API. It handles three things:

- Receiving the connection and replying/pushing messages to users
- Creating the different types of messages and objects to send to users
- The question bot structure of queuing up responses to users responses in a way akin to a conversation; in order to handle building up more complex/organised data

It can also handle setting up new LIFFs and Rich Menus and multiple languages
Without extending, the question bot allows asking a question to the server as well as changing the language setting.

Iduco Bot

code/iducoBot.php

My first proper job in Japan was working at a takoyaki restaurant named Iduco. During closing, tasks included cleaning, counting the nights earnings and comparing it to the cash registers receipt and doing stocktake. This is recorded on a form which is photographed and shared to the LINE group along with any notes of importance for the management and other employees to view. At the time, I made a web interface to achieve the same thing but had the idea that a LINE bot may be more intuitive. It was around that time that I started looking into the LINE Messaging API. This is a basic realisation of that, extending the LINE Bot code listed above.

enCharge Bot

code/enChargeBot.php

enCharge is a fictional company for one on one tutoring, this is a supplementary chatbot designed to assist students and help engagement with the service. It has several features:

- *Schedule a Session* by choosing a time, location, topic and difficulty
- *Search the Dictionary* for translating words between English and Japanese, when a user translates a word; it saves to their account in the database
- Do Flashcards of the words that they've previously searched or learned in lessons
- A *Scratch and Win Ticket* where a user tries to reveal three matching symbols to try get discounts on lessons or void costs like travel or drink fees.

As it is more fleshed out than the Iduco Bot, all of the following code files are tied into the functionality of this bot.

enCharge Version History

code/enChargeVersionHistory.pdf

This is a history of the features implemented in the three alpha iterations based on user feedback from the HCI testing process. It also includes things to implement in future iterations of the bot, which I haven't done so as this is a minimum viable product for portfolio purposes.

PHP Core/DB

code/core.php
code/db.php

This is a minified version of my core PHP functions and classes that are used for communicating with the frontend as well as a wrapper for accessing the database. If you would like to see an unminified version with some documentation, please see my general portfolio code on https://randomcode.run

Scheduler LIFF

code/scheduler/sceduleLIFF.html

A basic week view calendar that allows users to view what timeslots are already booked as well as allowing the user to visually select a timeslot rather than using the LINE Messaging date-time selector interface which may be more intuitive for the functions purpose. This is a simplified version of a web interface that I made for achieving a similar functionality. As the LIFF system doesn't allow me to call external files into the browser window, I've included a minified version of my JS core file in this and the other included HTML files. The way I've got my server handling serving pages is quite dynamic and the core has some really interesting functionality. It's using vanilla JS but acts as a personal code library that I'm more than happy to talk about in an interview.

Fetch Week

code/scheduler/fetchWeek.php

A simple script to fetch all the sessions of a given week.

Scratch And Win LIFF

code/scratchAndWinLIFF.html

One of the users had given feedback about something novel for using gold earned from doing Flashcards in the enCharge bot. It uses touch or mouse interaction to simulate scratching a lottery ticket. The original version of this was inspired by a minigame from a Nintendo DS title that I found novel and decided to implement.

Ticket Symbols

code/scratchAndWin/ticketSymbols.php

In the original scratch and win game that I made, random symbol generation was done in the Javascript. However as this is designed as a customer facing game with monetary rewards, I decided it was best to minimise cheating the system to get rewards. The randomisation is based on a key given when the customer “purchases” a ticket so they can’t arbitrarily open tickets until they win. The results of the symbol generation is stored in the database so the user can’t tinker with code to get whatever symbols they want. The randomisation is based on percentage based probabilities so the server can set the likelihood that certain prizes are achieved.

Question Ticketing System

code/questions/ticketer.html

The ticketing system is for handling user queries from the Question Bot and its extensions. It organises queries into open and closed tickets. When a user asks a question, it opens a new ticket. It’s based off a basic chat system I created to try implement PHP SSE. Other than the added ticket functionality, I also repurposed the friends list to show the staff members open claimed tickets. The remainder of code files are PHP helpers in aid of this system.

Chat SSE

code/questions/chatSSE.php

A PHP Server Sent Event file designed to send chat updates to the question ticterter web interface. SSE is only one way communication, like a radio broadcast. However, in this implementation of chat two way isn’t necessary as when a message is sent via LINE or the web interface it is stored in the database. This file checks for any relevant updates to the database and then sends them to the page. Though, SSE can be a little slow; it suits the purpose of chat and is more efficient than other techniques like long polling or comet. A better solution for more rapid exchange of data(ie a game or video chat) would be to run a node.js based server. But that is beyond the scope of what I wanted to achieve.

Claim Ticket

code/questions/claimTicket.php

When the staff member claims a new ticket, it calls this file to register the claimed ticket to the database. Multiple staff members can be on a ticket. A prerequisite of being able to respond to users is to have claimed the ticket.

Close Ticket

code/questions/closeTicket.php

Close the ticket with the flag of being successful or unsuccessful.

Fetch Users Tickets

code/questions/fetchUsersTickets.php

Get a list of the tickets that a staff member has currently open in order to populate the old friends list and allow for quicker messaging to multiple clients simultaneously.

Get Ticket Chat

code/questions/getTicketChat.php

Get the chat history of a conversation.

Get Users Questions

code/questions/getUsersQuestions.php

Get a list of tickets associated with a user whether open or closed in order to see what they've asked in the past.

Question Tickets

code/questions/questionTickets.php

Get all a list of all the tickets for the staff to open and view, can be filtered by the tickets status.

Send Message

code/questions/sendMessage.php

Send a message from the staff via web interface, it stores in the database and calls the sendToLine php file.

Send Support Ticket

code/questions/sendSupportTicket.php

Receive a message from the user via the chat bot. If there is no open ticket, it will

Send To LINE

code/questions/sendToLine.php

Push a message to a users LINE account.

Share Ticket

code/questions/shareTicket.php

Share or pass a ticket along to another staff member.

Images

images/

A collection of the various images used by the chatbots.

HCI

Personas

hci/personas.pdf

A collection of personas and scenarios created for stereotypical users.

Usability Template

hci/usabilityTemplate.pdf

Blank English and Japanese consent forms, observation sheets and questionnaires.

Usability Results

hci/usabilityResults.pdf

A selection of various results from usability tests. Though there were 16 tests, I've cherry picked the 6 that had interesting feedback or unique demographics. The results were retyped from the shorthand handwriting.

Paper Prototype Images

hci/images

A collection of feedback of things to change based on user feedback.

Translation

Narrow Cast Messaging

translation/narrowCast.pdf

In January, I noticed a new untranslated section of the LINE Messaging API. Although, since then it has been translated; at the time I thought it was worth translating. I haven't compared this to the official documentation, but I decided it was worth including as it shows my Japanese ability.

Root

Cover Letter

coverLetter.pdf

The cover letter in Japanese and English..

Rirekisho

rirekisho.pdf

This is the Japanese styled resume attached with my application.

Ideas For LINE

ideas.pdf

This file is a page of various ideas that I could bring to the table in order to improve the LINE service in the event that I do get hired. I figure it's not a normal thing to include in a portfolio, but is novel

Read Me

readme.pdf

The current file explaining the other files in the portfolio.

Thank you for taking the time to look through my LINE portfolio. If there are any discrepancies or questions, please feel free to ask. ***Disclaimer****: This was done in my free time when I wasn't doing work for my previous job, on a Chromebook with a half working mobile phone and no budget. It's lacking a little polish, but I can guarantee if given the chance; I will work to my highest possible level of excellence.*

I look forward to hearing from you.

-Matt Bird