**Notes on Internal Ticketing App**

* Plan out everything before beginning to code
* Organize features

Green = find out.

Red = more important feature

* Definitely possible to have it be email integrated.
* Make types of issues (Computer, Printer, Media device, so on)
  + I’ll have to make a comprehensive list later. (View old tickets)
  + Add a “Ticket” type incase type is not clear
* Add feature where duplicate tickets get an automatic response.
  + Hopefully this feature will be able to reply to senders with the previous solution we used.
    - OR possibly create answer templates that we can choose from.
  + This feature should only be implemented for simple problems (e.g. troubleshooting PC that won’t turn on)
  + Organize information on this feature more before beginning the code.
* Automate parts of the reply email. (e.g. “Consider this ticket closed” can be taken care of, along with other strings)
* Weekly email sent out of ongoing issues and how to temporarily bypass them
  + With weekly email, remind coworkers on how to send ticket emails
  + Create a status to see if email was sent out (viewable in app)
* Need to store all these tickets.
* Create accounts so we know if someone is responding to a ticket
* Allow option to fill in signature automatically (hopefully with formatting)
* Closing statement filled in automatically “Consider this ticket closed”
* Allow option to classify something as a ticket or not
* See if can get email chain to appear on the app.
* Ticket search options for techsupport users
* Allow option for certain automatic emails like “On my way” or “resolved” or “Couldn’t solve” and so on
* Reminders to reply to tickets
* Automatic sign-in
* Create roles with specific permissions
* Settings for notifications
* Modularity (what did you mean, past Chris?)
* Important Info/images section. (kinda like personal notes). (consider if it should be shared or individual)
* Keep track of when a solution email was last used
* Project tracking somehow
* Refresh tickets when scrolling up from the top

**FLOW**

1. Ticket gets sent via email to techsupport
2. App checks to see if there has been a similar previous issue and offers automatic response (for certain relatively simple tasks). Coworkers try to solve the issue and respond yes or no. (“Does this describe your issue? If so here’s a potential solution, otherwise please reply with “Wrong issue”. [Solution]. If this solution worked for you please reply with “yes”, otherwise reply “no” or “Didn’t attempt”. Thanks”) Maybe in the beginning can just offer techsupport some automated responses to send, otherwise just allow to send an email either from the app or be redirected to an email app. (redirection is easier to implement but more time consuming or annoying for techsupport users).
3. If not solved. Techsupport goes to help
4. Techsupport updates app to see if should be marked ongoing or completed
5. If completed techsupport responds (Try to make this easy through the app), ticket is marked completed and stored. If ongoing, ticket updated in the app and response goes out stating that it’s ongoing and listing any temporary fixes.

**Expanding on Flow**

Solution emails = problems we’ve solved in the past and described in detail

1. Receiving ticket via email:
   1. Need to connect outlook using JavaMail API (for sending and receiving)
   2. Check if can reply from my own work email
2. Suggesting previous solutions
   1. Implement a mysql database to keep desired solution emails
   2. Search based on ticket ~~subject~~ body. (match with solution email subject and/or body)
      1. Expand on this
   3. Reply with solution email’s subject, description and solution.
3. Suggestions didn’t work
   1. Have them reply with a certain phrase in subject line (e.g. “Didn’t work”)
4. Techsupport to the rescue
   1. If problem persists, mark it as ongoing
   2. If problem is solved mark as complete in app and reply with detailed solution
5. Completed
   1. Solution email created in detail and is sent as a reply with certain phrase at top of email body. (e.g. “Ticket closed”)
      1. Auto send “ticket closed” email when closing tickets but not replying.
   2. Keep email solution in a waiting list to be either appended or disregarded to the solution emails list.

**Ticket details:**

Need to create ticket objects. Attributes like open, closed, ongoing and so on (more details below)

List ticket attributes (for good tracking). e.g. Department, Person who solved, current status, etc (and afterwards label the, e.g. printing issues, monitor issues, so on)

Ticket statuses: open (red), auto-responded (yellow), ongoing (blue), closed (green)

**Filtering solutions:**

After completing ticket, ask if that solution should be added to database. (only ask users with certain role. I’ll have that role in the beginning and add it to anyone else if need be.). Organize solution by type (printing, public PC, etc)

**Project Tracking:**

???

**To-do**

* Break down “flow” further
* Sketch UI, and relationship between each screen (wireframe)
* Get main functionality working. (Receiving emails and displaying them on the screen)
* Then work on replies
* Get the UI/UX done for all pages

**To-do slightly later**

Microsoft Graph API has delta query. See if that will help with continuously updating the app

Authentication

Integrate Analytics

How to secure the app