- Slack API
  - https://api.slack.com/events/api
    - private channels: https://api.slack.com/events/message.groups
    - files [snippets / files]:
    - 

- GitHub API
  -
- Technology
  - SQL
    - relational data (events modify channels / teams / users)
  - Node
    - Python I/O
    - Express
- **Roadmap**
  - **AutoBot**
    - **Passive Systems**
      - Slack listening
        - event: message exchanged **[channels / groups / im / mpim]**
          - type of message
          - length of message
          - number of messages exchanged per channel / team / user can be incremented off this call
        - files added **[files / snippets / google docs]**
          - lines of code
      - Ticket system
        - slash command or a keyword to call the bot with the form link
          - make a form and have it post to the ticket database
    - **Active Systems**
      - team creation and populating

    - Sorting Hat sends a list (per cohort / team) of users and their teams and partners
    - creates channels
    - places users in their respective channels
    - Send a “getting started” message(s)
        - Screencast / show everything to do in Chingu
        - ice breaker questions
        - milestones (per tier / team)
        - send private interactive messages to each user that confirms:
            - they have been placed in their team
            - they have reviewed the getting started messages
            - they have communicated with their team and are ready to begin
- warning
    - users’ score falls below threshold
        - private message checking in with the user
        - confirmation interactive message that they have received the warning
        - record and attach all responses to that warning ticket (as threaded responses)
        - attach users’ score at the time of the warning
        - for testing send these messages to Chance
            - Chance manually confirms that the bot was correct / incorrect and tweaks accordingly
    - team misses milestone(s)
        - message the team and indicate which milestone was missed and its deadline
            - depending on which milestone is missed determine which message to send or action to take
            - record and attach all responses to that warning ticket (as threaded responses)
            - attach all users scores at the time of the waring
    - final warning
        - user / team receives a final warning that they will be removed or the team disbanded

- user removal / team disbanding
    - the team / user should be reported to chance
        - include all previous warnings / interactions
        - Chance reviews the situation
        - interactive message with approve / cancel the removal or disbanding
            - Approve: bot removes / disbands
            - Cancel: Chance handles it personally
- encouragements
    - users above threshold score
    - teams delivering ahead of milestones
        - receive encouragements and praise for their work
        - receive leaderboard data etc
- reminders
    - channel / team / user specific
        - milestones are due
        - user hasn't updated a project in a while
        - projects showcase coming up
        - beta tests
        - any other upcoming events / deadlines
- completed projects
    - send info to dashboard
    - send team / users / repo into project showcase channel (chingu central and per cohort)
        - tag teams / users to add info in a threaded response
- project showcase channel
    - autobot moderates and only allows responses as threads to each project
    - warns / deletes non threaded responses that don't originate from Chance or AutoBot

- **Admin / Dashboard**
    - Public
        - When accessing the admin / dashboard page without authentication (Chance)
            - land on an instructions page for the API to request all of the public data
    - WebApp
        - Charts (averages / outliers)
            - Cohorts
                - average score of all users

- average project completion % for all teams
- top / bottom 10% of users and teams
- top comments / links / messages / files of the day / week
    - based on threaded responses
    - reactions
    - shares
- completed projects
    - have users tag or upload their projects
- premade messages (reminders / encouragements / warnings)
    - custom messages from Chance sent by the dashboard
        - send from the dashboard to the autobot which relays into Slack
        - signed by Chance

- At a glance (all teams)
    - Cohort
        - color coding and sorting of all the teams
            - green / yellow / red
                - slack data and milestones
            -
- Ticket system (sorted by type with tabs)
    - warnings
    - removals / moving members from Autobot
    - general tickets / issues
    - have preset or customizable responses that are relayed through AutoBot
- Completed projects
    - profile bot pulls each weeks' completed projects
        - attach teams / users / github repo

- GitBot
  - Structure
    - Cohort Organizations on GitHub
      - Each team has a repo
        - master branch (local testing master)
        - production branch
          - protected requiring all team member approvals on PRs before merging
  - Passive Systems
    - Quantitative data
      - commits
        - per user and per team
          - number of commits
          - LoC associated with commit
      - issues raised
        - when working on a branch if you see another problem to work on raise an issue but finish working on that branch first
      - pull requests / merges
        - pull requests
        - approved pull requests / merges
        - LoC associated with merge
          - user
          - team
  - Active Systems
    - Setup
      - Chance would make a Cohort organization and link the GitBot
      - Creates repos for each team
      - Invites each user to each team
      - Creates master and production branches
        - protects production branches
      - Set up project
        - Set up milestones
        - Set up the project cards
    - Issues
      - as issues come up the bot raises an issue in the team's repo
    - Merges
      - milestone merges

        - tag milestones in the merge messages and trigger a bot listen event
        - user / team update merge count and milestone progress

    - Sorting Hat
        - selecting project managers
            - previous project completions (in chingu or otherwise)
        - send Chance a list of the selected tiers / teams / partners and their associated scores
            - reviews the list
            - modify incorrect placements
            - send final list to autobot for creation / population of channels and teams

Milestones

1. Tier 1
    1.1.
2. Tier 2
    2.1. start at step 10
3. Tier 3
4. Tier 4

ACT I

Everyone in and has sent a message
Picked a Project
Scheduled a meeting
Had successful meeting
User Stories Set
Tasks Assigned
Github repo logged

/autobot-resources

ACT II

ACT III

General Ideas

- cutting the accepted users in half to increase completion rates

- front end certification at minimum
    - or equivalent

Michael's Notes

1. Create data model
2. Outline responsibilities
3. Assign behaviors
4. Code the hell out of it

api
front end

sorting hat bot -> group assignment / repair / pair programming / accountability / freeagent
chingu chimp -> user profiles / user grading
elon musk(chance bot) -> reminds people, kicks people
autobot -> (get's information from sorting hat grading) cohort creation, voyage team creation

voyage team creation

1. assigning project manager