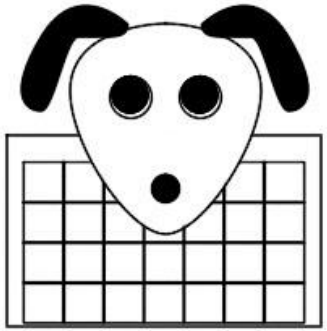


F.I.D.O.

The new scheduling application for the CAT

The Project Statement

- Computer Action Team needed a new scheduling system for DOGs
  - Desk CATs
  - Old one was low on features
  - admin had little control, no ui

- Our solution: FIDO
  - Fully Integrated DOG Organizer
  - CSS and Javascript make it beautiful and more powerful

Project Management

- **Agile flavour: Kanban**
  - **No sprints, just “drink from the fire hose”**
  - **Restrict how many stories can be in a specific swim lane at a time**
    - **Creates bottlenecks on purpose to keep the team moving**
- **Use Trello to organize cards**
- **Engineers did both sides of development (Developer and Quality Assurance)**
- **Team members also had administrative rolls from last term**
  - **Architecture, Communications, Specifications, Product Owner, Infrastructure, Risk Management**

Schedule

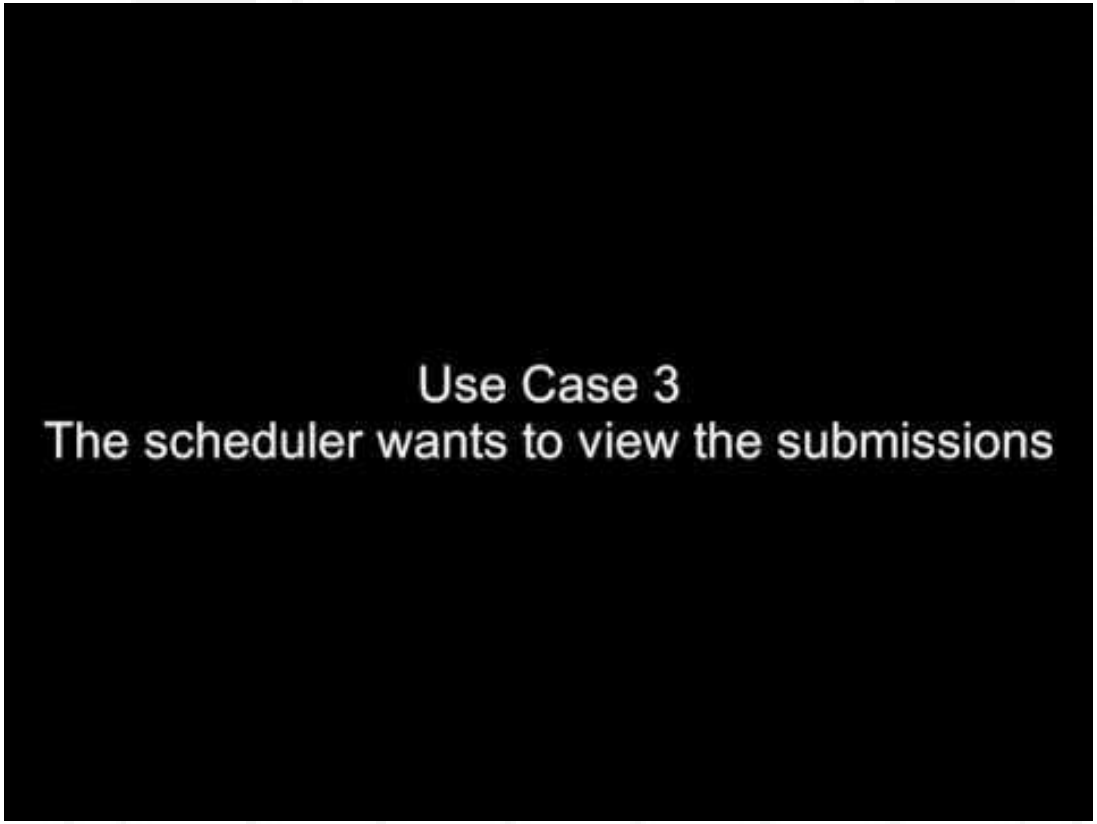
Initial Plan:

- Week 1: Backlog Grooming.
- Week 2-3: MVP Implementation.
- Week 4-5: Beta Testing.
- Week 6-9: Stretch Goals.
- Week 10: Create Final Presentation.
- Week 11: Sponsor Delivery and Final Presentation.

What we Did:

- Week 1: Backlog Grooming.
- Week 2-5: MVP Implementation.
- Week 6-9: Beta Testing and Stretch Goals.
- Week 9.5: Code Cleanup.
- Week 10: Create Final Presentation.
- Week 11: Sponsor Delivery and Final Presentation.

DEMO



The Architecture

- Tech Stack
  - HTML
  - PHP
    - Server side computation
    - Database interface
  - JavaScript + JQuery
    - Client side computation
    - Dynamic modification of DOM elements
  - PostgreSQL
  - CSS (Bootstrap)
    - User interface
- Organization
  - Queries
    - Organized by database table
  - API
    - Organized by user
  - Pages
    - Organized by user
  - JavaScript
    - Mirrors pages

The infrastructure

How did we manage our source code?

- Github was the primary host for our code outside of the CAT Stash Backups
    - Our git flow had a couple models which changed with the steps in version
    - During the development process prior to stretch goals when we constructed our MVP
        - We had a rather simple model of a dev branch and a master branch
        - Devs would push their to be tested code to dev and at QA Done pulled to master
    - After this we realized limits to this and the higher chances and difficulty with conflict
    - We switched to a model where devs had their own branch
        - A branch would represent a feature and be unique to the dev
        - The branch would get a pr to pull to master upon being complete for QA
        - It would be pulled into the master (stable) when QA Done

Where were our backups, and how were they maintained?

- We used the CAT's stash as a way to hold versions of our code
    - The CAT provides weekly backups for the stash
    - We also had weekly or bi-weekly (two times a week) backups stored in this stash
    - Backups were pulled from dev and the master prior to the change and then master

Risk

Problems and Solutions

- Code conflicts
  - New branch for each Story/Card
- Communication
  - Different channels in Slack
- Lack of knowledge
  - Pair programming
- Transparency
  - Google drive for documents

Risk

How did the schedule of the project help mitigate risk?

- Code conflict
- Dependencies

What were the critical parts of the project, and how did we ensure they were completed to mitigate risk

- Integration of different pieces of code
- Meeting the minimum requirements

Documentation

- **Documents Created**
  - Work Breakdown Structure
  - Stretch Goals
  - Meeting Notes
  - Requirements
- **Code Organization to Facilitate Maintainability using Comments and Documentation**
  - First we separated code into corresponding folders and files
    - As seen in the architecture slide
  - All functions commented and made in same style
    - Most functions would use the same or similar variable names
      - Functions have descriptions, parameters taken, return values, and any notes
- **Code Cleanup**
  - Finished up most of stretch goals then went through all the code to make sure everything we had was necessary
    - Also during this process we checked that all code was adequately commented for future maintainability

Communications

- Primarily used Slack and email
- Met with sponsors during our weekly team meetings, messaged them via Slack as questions came up
- Alex would talk with the scheduler and beta testers in person
    - Went through live walkthroughs, feedback was compiled into reports and made available to everyone in the team
- Having the sponsors available on Slack helped keeping the development progressing smoothly
    - Clarifying requirements
    - Design feedback/suggestions
- Everybody would respond to messages / announcements in a timely matter
    - Worst delay was about a day in a half

Conclusion/ Take away

What we learned

- Alex Simchuk
- Cody Wyatt
- Nima Sajadpour
- Isaac Archer
- Graham Drakeley
- Eiyad Alkadi
- Jonathan Castle