Team 3: Smart Scheduler Bi-Weekly Update 4

Collin Bennett, Nicholas Garde
Sponsor: Narasimha Annapareddy

TA: Eric Robles

```
        ekday

    every list [date,[times]] in empties
 pties:
  the date in empties is the same as the date of the
  make sure that we only effect times on the day the e
if k[0] == evDate:
    l = 1
    # iterate through the times of k and change
    while l < len(k):
        # FIRST if the event is the same a
        if (j.sTime == k[l][0] and j.eTi
            k.remove(k[l])
            # drop l down by one a
            l-=1
        # THEN if the event li
        elif (j.sTime > k[l]
            temp = k[l][1
            k[l][1]=j.
            k.append
        # THEN if
        elif (j
            
        #
```

Introduction

- Scheduling meetings can be very difficult, so we are simplifying the process.
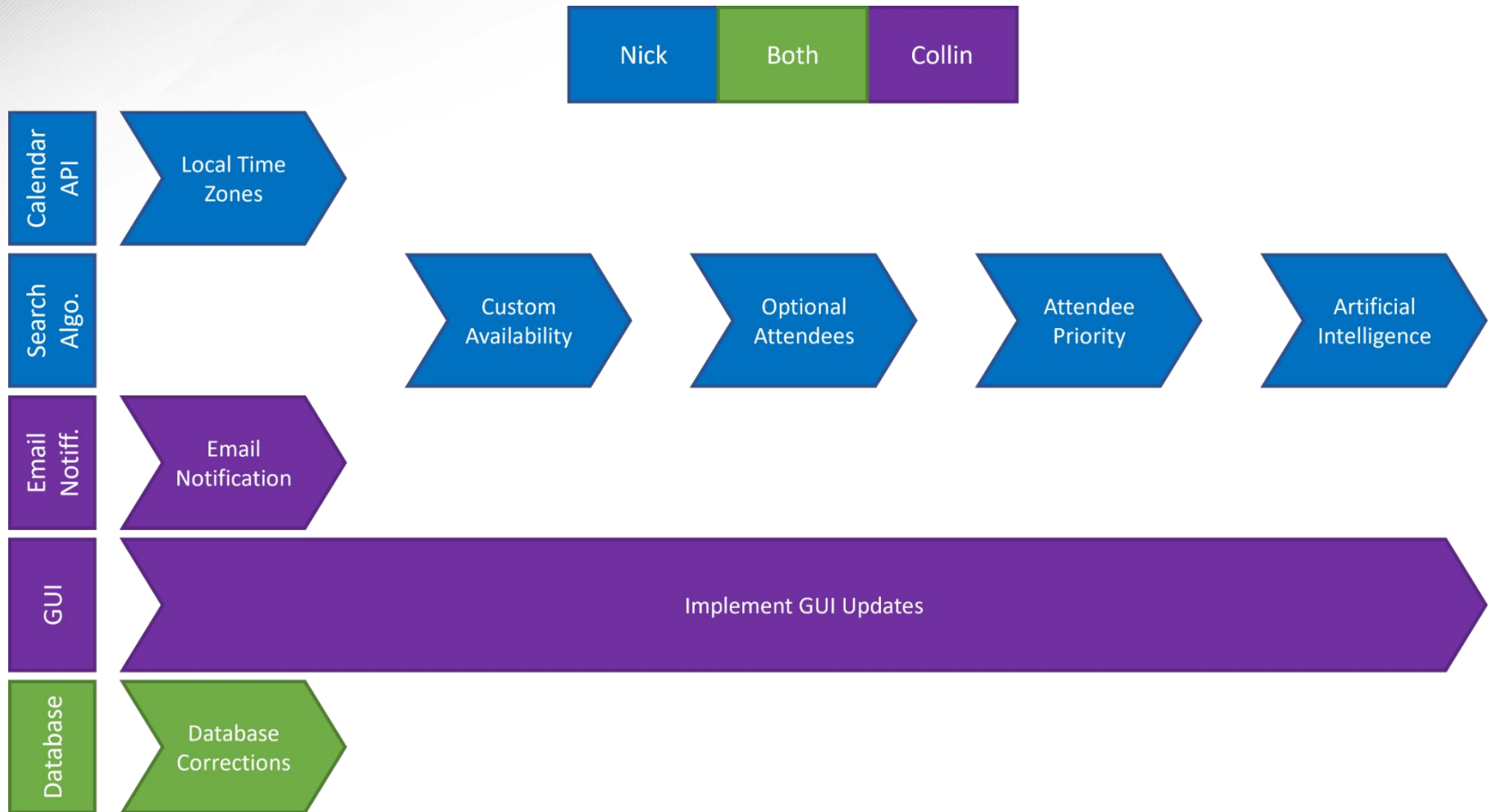
Mon	Tue	Wed	Thur	Fri

Project/Subsystem Overview
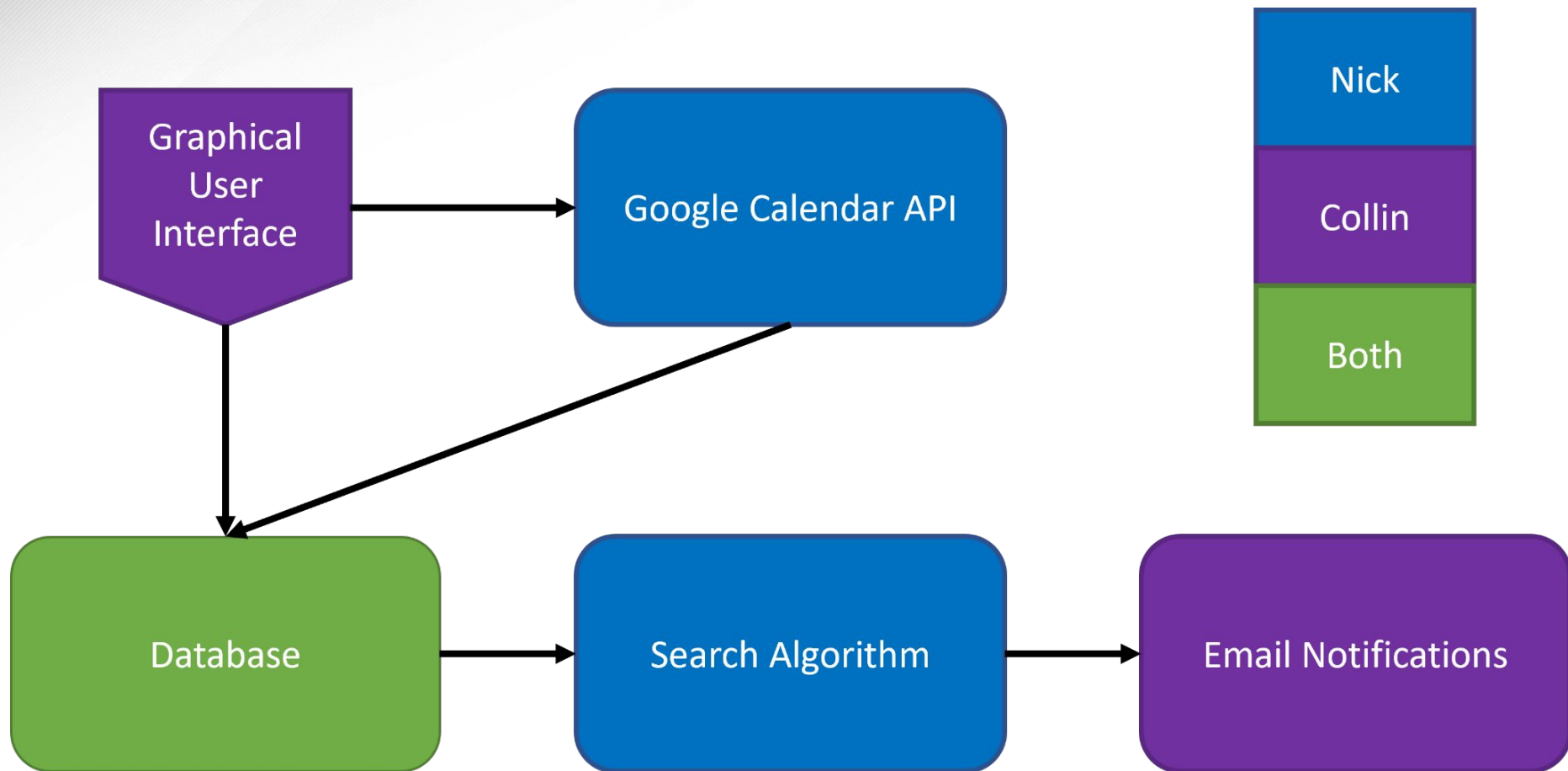
Graphical User Interface

Owner: Collin

Since last presentation: 10 hours of effort	Ongoing progress/problems and plans until the next presentation
- Added custom start and end times - Deleted RSVP page (simplified to pressing link in email)	- Implement optional/required attendees

Graphical User Interface

Owner: Collin

- Planning to add the ability to import contacts from Google
- Making updating and deleting contacts simpler

After

Before

Database

Owner: Both

Since last presentation: 5 hrs of effort	Ongoing progress/problems and plans until the next presentation
- Database now holds user timezones, custom start and end times, and attendee optionality	- Slight modifications/refinements will be made as needed

Search Algorithm

Owner: Nick

Since last presentation: 20 hrs of effort	Ongoing progress/problems and plans until the next presentation
- Developing methods of scoring priority and optional attendees - Bug-fixing data structure and results arrays	

Search Algorithm

Owner: Nick

Current Design

- Places events onto calendar data structure
- Searches by # of conflicts
- Take into account number of participants in each meeting
- 24hr data structure
- Optional Attendees
- Take into account priority of participants
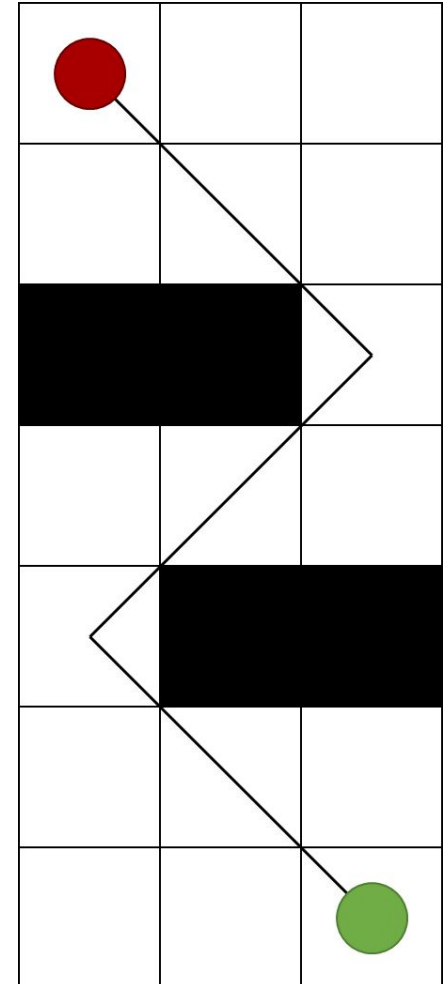
Calendar API

Owner: Nick

Since last presentation: 15 hrs of effort	Ongoing progress/problems and plans until the next presentation
- Optional "Remember Me?" - Ignoring JSON file system	- Hosted bug fixes

Calendar API

Owner: Nick

Current Design
- Pulls user events from their calendar
- Pushes created events back onto their calendar
- Retrieve user time zones
- Option to skip authentication

Planned Design
- Hosted Functionality

Email Notifications

Owner: Collin

Since last presentation: 10 hours of effort	Ongoing progress/problems and plans until the next presentation
- Email notifications have been integrated into the app (not as a branch) - Emails have been styled (changes not yet integrated)	- Continuation of testing and refinements

Email Notifications

- Need to integrate stylings

Meeting Invitation

smartscheduleralerts@gmail.com
to me ▾

You have been invited to a meeting. Here are the details:

Meeting name: Demo Meeting
Description: Testing Functionality
Deadline: August 03, 2023
Duration: 3 hours and 30 minutes
Link/Location: Texas

To RSVP, click the following link:
localhost:8000/accept_meeting_google/57/46/accept

To decline, click the following link:
localhost:8000/decline_meeting/46/decline

Meeting Confirmation

smartscheduleralerts@gmail.com
to ▾

You have been confirmed for the meeting. Here are the details:

Meeting name: Demo Meeting
Description: Testing Functionality
Time: 1300
Duration: 3 hours and 30 minutes
Link/Location: Texas

View this event in your Google calendar:

https://calendar.google.com/calendar/u/0/r

Execution Plan

Subsystem	August 31st	September 14th	September 28th	October 12th	October 26th	November 7th
				Completed	**In Progress**	**Not Started**
Search Algorithm (Nicholas)	Review changes to algorithm and receive sponsor feedback	Implement timezone functionality	Implement custom start and end times, valid days	Researching better machine learning and lowest-cost solutions	Iron out any potential oddities in the search algorithm	Have search algorithm completely done
Database (Both)	Move from firebase to relational database	Simplify database, getting rid of unnecessary tables	Have database ported from SQLite to PostgreSQL and hosted.	Continue work on other subsystems	Check database for correctness	Database is done, hosted and complete
GUI (Collin)	Django website bug fixes and refinements	Have some form fields optional to speed up the process	Have required and optional attendee functionality integrated	Get additional user feedback on the website style	Implement changes based on the user feedback	GUI is finalized and styled
Email Notification (Collin)	Rework email to SMTP from API functionality	Email is properly styled	Email is fully functional on a basic level	Email is not being sent to user's spam folder	Validate and test email	Email is sending without error and finalized
Google APIs (Nick)	Review and research Google APIs in Django	Integrate timezone functionality with Google API	Integrate Google contacts imports	Begin testing correctness of API calls	Finalize testing of API calls	Google APIs are finalized

Validation Plan

Task	Specification	Method	Result	Owner
Google Calendar - Pull Contacts	Test Fetching Accuracy (No errors)	Using Google's OAuth2.0 to request contact data from the user from their email	Pending	Collin Bennett
Google Calendar - Insert Event Times	Google API is properly inserting correct event times	Ensure that calendar events can be added to specified user calendars without error. Specifically, ensuring that users in different time zones are being sent the proper information.	Complete	Nicholas Garde
Database - User Authentication	Database is storing user account details	Ensure PostgreSQL solution is working properly	Complete	Collin Bennett
Database - Event Parameters	Ensure event parameters are added properly	Add several events of varying nature to Database from the GUI, ensuring that they are properly added each time by comparing the inputted values to those stored in Database.	Complete	Collin Bennett

Validation Plan (Continued)

Task	Specification	Method	Result	Owner
Database - User Events	Ensure User events are properly added	Pushing each user's calendar to database to be stored until the data is needed for the search	Success	Collin Bennett
Search Algorithm - Time Zones	Ensure User events have proper times	Ensure that users in different timezones are correctly accounted for by the algorithm	Success	Nicholas Garde
Search Algorithm - Additional Parameters	Valid times	Ensure that additional parameters are correctly accounted for by the algorithm	Success	Nicholas Garde
Email API Correctness	Testing of Email Notification Scenarios	Ensure the email link clicks make the proper writes to the database and move efficiently.	Success	Collin Bennett
GUI - Ease of Navigation	Exploring GUI navigation ease-of-use	Ensure the GUI navigation works properly, nothing is out of place, and correct error messages are displayed.	Pending	Collin Bennett
Bug fixing	Attempting to break and Find Errors in Our Program	Ensure that the program works under all conditions by scheduling numerous programs under varying conditions and checking that they are scheduling correctly.	Pending	Collin Bennett and Nicholas Garde

Questions?