Links
Heroku: https://eventnxt-fall2024-demo3-c90f7940fb8e.herokuapp.com/
Linear: https://linear.app/eventnxt/team/EVE/all
Github: https://github.com/FashioNXT/EventNXT-Fall2024/tree/dev

Dates
28 October, 2024 - 7 November, 2024
Members Contributions

Project Manager	Duties		Estimated Time
Neil Gautam	Made user stories Check pull requests Communicated with client Made documents for sprint Made sure discussion assignments were done as a team Testing of the functionalities during the sprint and reported the consistent progress to the client during sprint Did rubocop fixes		10
Scrum Master	Duties		Estimated Time
Rishabh Pagaria	Daily check ins with developers Made documents for sprint Check pull requests Data Validation Features of Manage Invited Guest Made User stories Did rubocop fixes		10
Developers	User Stories	Points	Estimated Time
Ming			
	1. Feature: Show Box Office Sales Data from Eventbrite (Scenario a)	2	2 hour
	2. Feature: Show Box Office Sales Data from Eventbrite (Scenario b)	4	4 hours

	3. Feature: Show Box Office Sales Data from Eventbrite (Scenario c)	3	3 hours
Total		9	9 hours
Surya			
	4. Feature: Referral link	3	3 hours
	5. Feature: Sent email summary table	3	3 hours
	6. Feature: Unsent email summary table	3	3 hours
Total		9	9 hours
Parth			
	7. Feature: RSVP link	3	3 hours
	8. Feature: RSVP committed seats to be updated in seating summary and guest summary	3	3 hours
	9. Feature: Manage referral table to be connected with Eventbrite data	3	3 hours
Total		9	9 hours
Kiara			
	10. Feature: Seating Summaries Sold Seat Column linked to Eventbrite Booking	3	3 hours
	11. Feature: Manage Invited Guest - Data from client (Scenario a)	3	3 hours
	12. Feature: Manage Invited Guest - Data from client (Scenario b)	3	3 hours
Total		9	9 hours

Sprint Goal

After the completion of sprint 2, the major user stories were implemented into this sprint. Because according to the client they were really important. These improvements are meant for dealing with the box office data and the email service to handle the emails sent from gmail.

Sprint Achievements

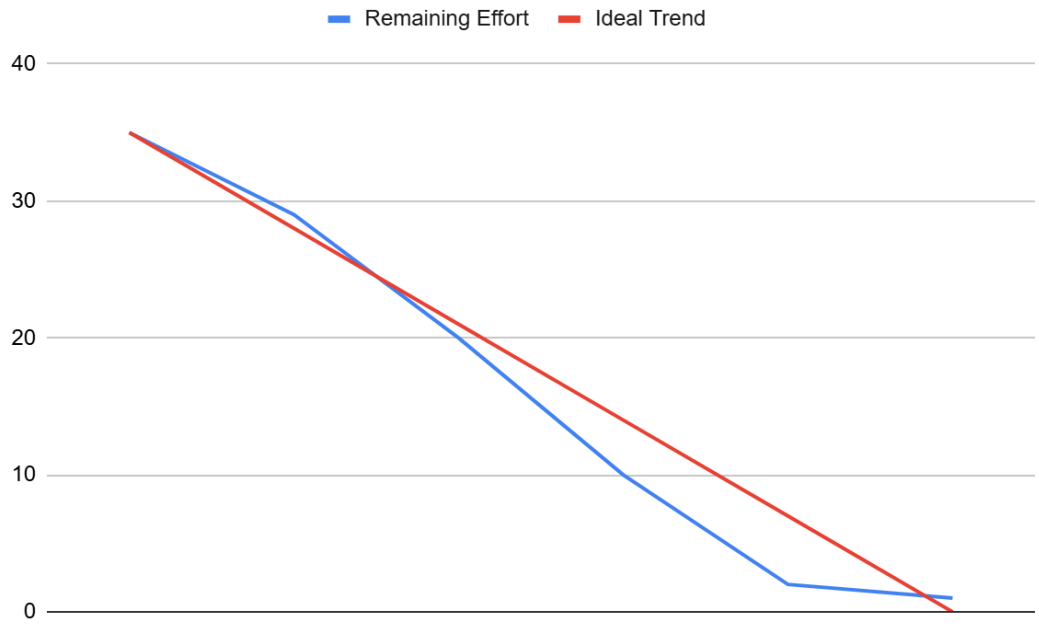
The team completed the majority of the user stories for the improvements and rspec fixes are completed. Sprint 3 achievements are as follows:

- Connecting the app through the Eventbrite external API for box office data.
- Login and logout feature implemented using the main events 360 CRM tool
- Email services like Referral rewards and RSVP invitations functionality enhanced,

Burndown

Sprint Burndown Chart						
User Stories	Initial Estimate					
	Day 0	Day 1	Day 2	Day 3	Day 4	Day 5
Manage Invited Guest - Data from client (Scenario a)	3	1	1	1	0	0
Manage Invited Guest - Data from client (Scenario b)	3	0	1	1	1	0
Show Box Office Sales Data from Eventbrite (Scenario a)	2	0	0	0	1	0
Feature: Show Box Office Sales Data from Eventbrite (Scenario b)	4	0	1	1	1	0
Feature: Show Box Office Sales Data from Eventbrite (Scenario c)	3	0	1	2	0	0
Referral link	3	0	0	0	2	0
Sent email summary table	3	1	0	1	1	1
Unsent email summary table	2	1	1	0	0	0
RSVP link	3	0	2	1	0	0
RSVP committed seats to be updated in seating summary and guest summary	3	0	0	2	1	0
Manage referral table to be connected with Eventbrite data	3	1	1	0	1	0

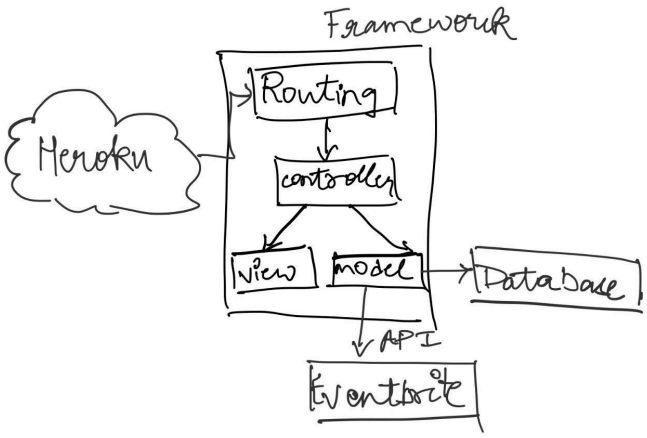
Seating Summaries Sold Seat Column linked to Eventbrite Booking	3	2	1	1	0	0
Remaining Effort	35	29	20	10	2	1
Ideal Trend	35	28	21	14	7	0



Documentation of Changes

- Removed the spreadsheet dependency for the Box Office Data.
    - After multiple meetings with the client, we had to come to the conclusion that using spreadsheets in the box office would not be efficient and would lead the app to break at some point because of the significant number of edge cases.
    - Number of negative edge cases which could lead the app to break because of using spreadsheets were significant, so we had to completely remove that functionality.

Design Diagrams



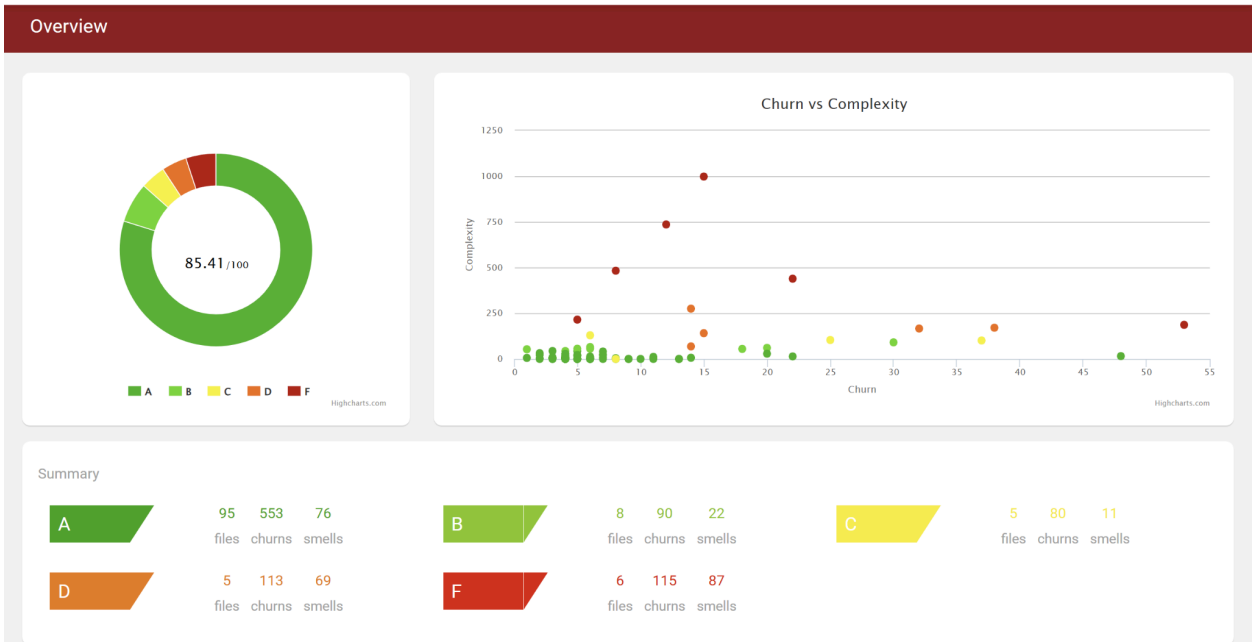
Customer Meeting

11/06/24

Customer was satisfied with the amount of work the team has completed so far. The customer still had a couple of concerns we are addressing. It was mainly decided to host everything into 1 branch and share the link with the client to test and give a sign-off.

Evaluations of Code and Test Quality

```
611
612  Finished in 5.42 seconds (files took 3.33 seconds to load)
613  211 examples, 0 failures
614
615  Coverage report generated for RSpec to /eventnxt/coverage. 695 / 767 LOC (90.61%) covered.
```



bdd & tdd

1. Eventbrite connect

```
  @omniauth_except_crm
  Scenario: Connect to Eventsbrite account                                  #
features/eventbrite_integration.feature:4
    Given I am on the event page "Eventbrite Test Suite"                    #
features/step_definitions/events_steps.rb:7
    And I have the following Eventbrite Events                              #
features/step_definitions/eventbrite_steps.rb:1
      | id | name |
      | 1  | First Event |
      | 2  | Second Event |
    When I click on "Connect to Eventbrite"                                 #
features/step_definitions/shared_steps.rb:13
    And I am on the event page "Eventbrite Test Suite"                      #
features/step_definitions/events_steps.rb:7
    Then I should see "Connected: Eventbrite"                               #
features/step_definitions/shared_steps.rb:30
    And I should not see "No events found."                                 #
features/step_definitions/shared_steps.rb:34
    And I should see the external events list showing "First Event, Second Event" #
features/step_definitions/eventbrite_steps.rb:11
```

```
+ require 'rails_helper'
+
+ RSpec.describe Users::EventbriteController, type: :controller do
+   describe 'POST #disconnect' do
+     let(:user) { nil }
+     let(:config) { instance_double(TicketVendor::Config) }
+     let(:service) { instance_double(TicketVendor::EventbriteHandlerService) }
+
+     before do
+       allow(TicketVendor::Config).to receive(:new).and_return(config)
+       allow(TicketVendor::EventbriteHandlerService).to receive(:new)
+         .with(user, config)
+         .and_return(service)
+       allow(service).to receive(:disconnect)
+     end
+
+     it 'calls disconnect on the service and redirects to events path' do
+       post :disconnect
+
+       expect(service).to have_received(:disconnect)
+       expect(response).to redirect_to(events_path)
+     end
+   end
+ end
```

```
describe 'eventbrite' do
  let(:user) { create(:user, Constants::Eventbrite::SYM) }
  before do
    # Mock OmniAuth response
    request.env['omniauth.auth'] = OmniAuth::AuthHash.new({
      provider: Constants::Eventbrite::NAME,
      uid: '123456'
    })

    # Mock the from_omniauth() method in the controller
    allow(User).to receive(:from_omniauth).and_return(user)
  end

  context 'when user exists and is persisted,' do
    it 'show notice' do
      get Constants::Eventbrite::SYM
      expect(flash[:notice]).to be_present
      expect(response).to redirect_to(events_path)
    end
  end

  context 'when user does not exist' do
    before do
      allow(User).to receive(:from_omniauth).and_return(nil)
    end

    it 'show alert' do
      get Constants::Eventbrite::SYM
      expect(flash[:alert]).to be_present
      expect(response).to redirect_to(events_path)
    end
  end
end
```

2. RSVP Link

```
Scenario: Render RSVP email template with generic placeholders # features/email_placeholder_replacement.feature:6
  Given I have an email with subject "RSVP Invitation"         # features/step_definitions/email_steps.rb:4
  When I render the email template                             # features/step_definitions/email_steps.rb:9
  Then the output should contain "EVENT"                       # features/step_definitions/email_steps.rb:13
  And the output should contain "FIRST_NAME"                   # features/step_definitions/email_steps.rb:13
  And the output should contain "LAST_NAME"                    # features/step_definitions/email_steps.rb:13

Scenario: Render general email without placeholder replacement # features/email_placeholder_replacement.feature:13
  Given I have an email with subject "General Email"           # features/step_definitions/email_steps.rb:4
  When I render the email template                             # features/step_definitions/email_steps.rb:9
  Then the output should contain "Original Body"               # features/step_definitions/email_steps.rb:13
```

```ruby
require 'rails_helper'

RSpec.describe EmailServicesHelper, type: :helper do
  describe '#render_template_with_generic_placeholders' do
    let(:rsvp_email) { { subject: 'RSVP Invitation', body: 'Original Body' } }
    let(:referral_email) { { subject: 'Referral Invitation', body: 'Original Body' } }
    let(:other_email) { { subject: 'General Email', body: 'Original Body' } }

    it 'replaces placeholders with generic terms for RSVP template' do
      allow(File).to receive(:exist?).and_return(true)
      allow(File).to receive(:read).and_return('<%= @event.title %> <%= @guest.first_name %> <%= @guest.last_name %>')

      result = helper.render_template_with_generic_placeholders(rsvp_email)
      expect(result).to include('EVENT')
      expect(result).to include('FIRST_NAME')
      expect(result).to include('LAST_NAME')
    end

    # it 'replaces placeholders with generic terms for Referral template' do
    #   allow(File).to receive(:exist?).and_return(true)
    #   allow(File).to receive(:read).and_return('<%= @event.description %> <%= @event.datetime %> <%= @event.address %>')

    #   result = helper.render_template_with_generic_placeholders(referral_email)
    #   expect(result).to include('EVENT_DESCRIPTION')
    #   expect(result).to include('EVENT_DATE')
    #   expect(result).to include('EVENT_ADDRESS')
    # end

    it 'renders body as is for other templates' do
      result = helper.render_template_with_generic_placeholders(other_email)
      expect(result).to eq('Original Body')
    end
  end
end
```

3. Referral Link

```
Feature: Update the referral table with created referral data

  @pre_authenticated
  Scenario:
    # features/referral_table.feature:4
    Given I am on the event page "Fake Event"
    # features/step_definitions/events_steps.rb:7
    And I have the following seats
    # features/step_definitions/event_seats.steps.rb:1
      | category  | section | total_count |
      | 1         | 1       | 80          |
    And I have the following guests
    # features/step_definitions/event_guests.steps.rb:11
      | first_name | last_name | email | affiliation | category | section | alloted_seats | commited_seats | guest_commited |
      | xx         | xx        | y@yy  | xx          | 1        | 1       | 1             | 1              | 1              |
    When we visit the new page for the referral
    # features/step_definitions/events_steps.rb:13
    When we enter 'aaaaaaa@aaaaaaa.???' into the 'friend_emails' field
    # features/step_definitions/events_steps.rb:17
    When we click the 'Send Referral Invitations'
    # features/step_definitions/events_steps.rb:21
    Then there will be one additional referral tuple generated with expected attibute on the referee email with 'aaaaaaa@aaaaaaa
??' # features/step_definitions/events_steps.rb:25
    When we have a referral with 5 tickets bought
    # features/step_definitions/events_steps.rb:29
    When we visit the show page for this event
    # features/step_definitions/events_steps.rb:48
    When visit the edit referral page
    # features/step_definitions/events_steps.rb:52
    When we enter 10 into 'Input'
    # features/step_definitions/events_steps.rb:56
    When we click submit
    # features/step_definitions/events_steps.rb:60
    Then the reward value will be updated to 50
    # features/step_definitions/events_steps.rb:64
```

```ruby
describe 'update method for referral after we have a referral' do
  let(:event) { create(:event, user:) }
  let(:seat) { create(:seat, event:) }
  let(:guest) { create(:guest, event:) }
  it 'then we will have reward updated' do
    the_referral_parametrization = {
      email: guest.email,
      name: "#{guest.first_name} #{guest.last_name}",
      referred: 'aaaaaaa@aaaaaaa.aaa',
      status: true,
      tickets: 3,
      amount: 150,
      reward_method: 'reward/ticket',
      reward_input: 0,
      reward_value: 0,
      guest_id: guest.id,
      event_id: event.id,
      ref_code: guest.id
    }
    @referral = Referral.create(the_referral_parametrization)
    @referral.save

    the_referral_parametrization_updated = @referral.attributes.merge(reward_input: 10)

    put :update, params: {
      event_id: event.id,
      id: @referral.id,
      email: guest.email,
      name: "#{guest.first_name} #{guest.last_name}",
      referred: 'aaaaaaa@aaaaaaa.aaa',
      status: true,
      tickets: 3,
      amount: 150,
      reward_method: 'reward/ticket',
      reward_input: 10,
      guest_id: guest.id,
      ref_code: guest.id,
      referral: the_referral_parametrization_updated
    }
    @referral.reload
    expect(@referral.reward_value).to eq(30)
```

```ruby
      put :update, params: {
        event_id: event.id,
        id: @referral.id,
        email: guest.email,
        name: "#{guest.first_name} #{guest.last_name}",
        referred: 'aaaaaaa@aaaaaaa.aaa',
        status: true,
        tickets: 3,
        amount: 150,
        reward_method: 'reward percentage %',
        reward_input: 10,
        guest_id: guest.id,
        ref_code: guest.id,
        referral: the_referral_parametrization_updated
      }
      @referral.reload
      expect(@referral.reward_value).to eq(15)
    end
  end
end
```