CSCE 606: THE SPRINTERS

EVENTNXT FINAL REPORT

Project Summary

The main customer requirement was to improve the legacy application. The customer, Mr.Tito Chowdhury, encouraged us to explore other event management tools like Eventbrite, Evite, Ticketmaster, etc., to fill in the gaps in the current project. EventNXT is expected to be a well-rounded tool for event producers, organizers, promoters, and patrons. Currently, the portal addresses three main functionalities in the fashion industry through automation:

1. Box office ticket sales (which are imported from a spreadsheet generated by a third-party app).
2. Guest list and RSVP management to send event invites to attendees and receive RSVP responses through email.
3. Referral reward system for customer acquisition which tracks referrals.

This system combines both RSVP and ticket management into one portal. It also tracks referrals using a referral reward system. A user can manage event tickets, promotion strategies, and guest referrals. Event organizers and Event Managers are the stakeholders who are responsible for managing the Box office sales, Ticketing, and Seating. The changes during this semester included fixing the bugs present in the existing legacy code and implementing new features such as Box office integration, RSVP, Referral, etc. The future scope of this project would be to bring in a referral reward system that keeps track of the person who was referred by maintaining a system for Promo codes.

User Stories

Set up the Heroku account and deploy the project - Chore (no points) - Completed: A new account was created in Heroku and was integrated with the GitHub repo A Staging and Production pipelines were created in Heroku by fixing the existing build issues. A heroku.yml file was added to deploy code in a docker container and to create Heroku-Postgres as an add-on post which the staging server was deployed successfully. The property "config.assets.compile = false" was removed from the production configuration to fix the production server failure.

Ticket sales third-party sites research **- Chore (no points) -** ***Completed:*** The research was made on third-party ticket sales websites Ticket master and Ticket tomato to understand the design of the web application to get the API documentation.

Setting up Rails Credentials **- Chore (no points) -** ***Completed:*** Download vim for windows was downloaded and the rails credentials were added by Setting the Editor to vim. The local development.yml.enc file was deleted to avoid a cred decryption error.

RSVP Research* - Chore (no points) - *Completed: The research was made on third-party event management websites EventBrite and Evite to understand the design and working of the RSVP process.

Referral Website Research* - Chore (no points) - *Completed: The research was made on popular websites to understand the design of the email template and the working of the referral rewards system.

Event website and third-party integrations research - **Chore (no points) - *Completed:*** The research was made on modern features like the Inclusion of Promo codes, Real-time announcements of the Event, Event Analytics, etc. to check the feasibility of incorporating these features into the application.

Send bulk emails to users informing them of RSVP Expiry - 2 points - Completed: Bulk Email functionality was added to iteratively send emails to multiple event participants. A new checkbox was added to select/unselect all the guest rows in the "Manage Guests" section. An email template was created for RSVP expiry to be used for informing users that RSVPs are no longer accepted. A new button "Bulk Email" was added on the UI that pre-loads all the emails of selected guests, gives the option to select an email template, and then calls bulk email API.

Make seating levels table editable - 1 point - Completed: Action links 'Edit' and 'Delete' were added for each existing seating type. Clicking on Edit will make the row inline editable and will get updated on clicking the 'Update seat' button. Clicking on 'Delete' will ask for confirmation and delete the row. A create row option was added in a more intuitive form. The styling of the table was enhanced.

Editable seating levels

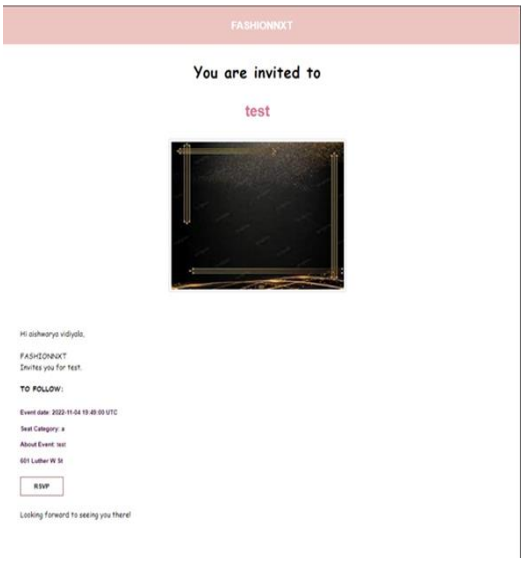
Check on the integration of RSVP and seating summary in the backend - 2 points - Completed: The discrepancies with respect to the seating summary table were identified. The allotted seat count was fixed to reflect the recent changes. The total guest count was fixed to take the count from the guests' table only when the allotted count is at least one. The Roo version was updated to upload the box office data.

Check SendGrid email API usage and how to add it to the current project - Chore (no points) - Completed: An API key was created to authenticate access to SendGrid services. The email address was set to the one given by the client on the SendGrid website to send emails to the guest from that mail id. The ownership of a single email address to use as a sender using Single Sender Verification was verified. A new .env file was created and port and domain parameters were set.

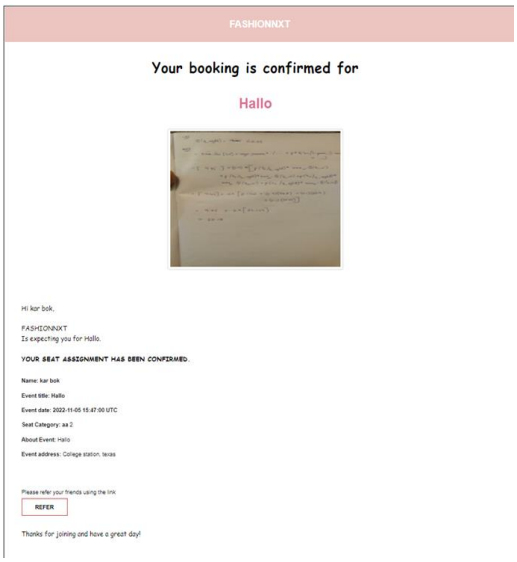
Load the data from box office sales dynamically and display on UI - 3 points - Completed: A new table “Sale Tickets” was added to store information related to box office sales. An input field was added on UI to take the header row number and display column values. A drop-down was created to select appropriate columns for the first name, last name, email, seat section & number of seats. A new section was added on UI to show sales information from the database.

Fixing the RSVP email template - 1 point - Completed: The styling of the RSVP email template was enhanced. New fields for seating and the number of tickets for the event were added. The event image was added to the RSVP email.

User Interface for the guest to provide RSVP, choose the number of tickets to be booked and store the details in the backend and fix RSVP Confirmation Email template - 1 point - Completed: Clicking the RSVP opens a new page where the guest will be able to see the seating level against the number of tickets field. The user can choose the number of seats he/she wishes to book in each level within the maximum number of seats. Clicking on Submit will save the data in the database and RSVP Confirmation Email will be sent to the user. The template of the RSVP Confirmation Email was fixed.



RSVP Invitation

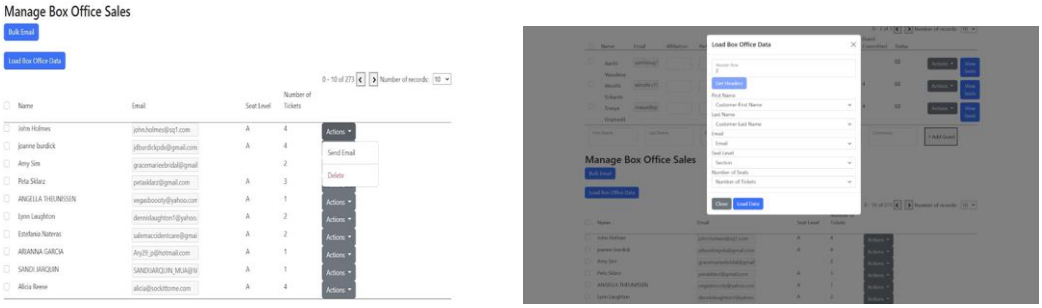


RSVP Confirmation Email

Expire RSVP link after given expiry 2 points - Completed: A button "Set invite expiry" was added to set invite expiry for multiple guests. The date and time can be set on the popup after clicking

the button. This updates the invite_expiration column for each guest selected and is also reflected in the Status column in the Manage guests section. If the guest opens the RSVP link after expiry, an appropriate message is displayed instead of the RSVP form.

Save the headers mapping for box office and use them for loading sales from a new file - 1 point - Completed: A new table was added in Postgres named “Boxoffice headers” to store information related to header mapping fields. A new function was added to store the header row numbers from UI to the database. A JavaScript function was added to retrieve the header information from DB. The dropdowns on UI were formatted to show the previously mapped header column to name mapping.



Loading box office data

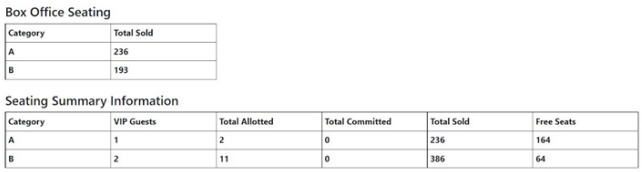
To add Perks and Comments columns in the Manage Guests sections - 2 points - Completed: New columns, Perks, and Comments were created in the Guests database. The Perks and Comments columns were added in the UI under the Manage Guests section. A functionality was added in the JS file to append the data for Perks and comments to the database.

Make the Name field in Manage Guests Sections editable - 1 point - Completed: The first and last names were changed to input type to make it editable by adding functionality in the JS file to update the first and last names in the Guests database whenever an edit is made.

Seating Allotment display at the guest level - 2 points - Completed: A button was added to view the seating allotments for each guest. A JS function was created to fetch the seating level allotments. A view was created to display the seats allotted with the categories and counts for each level.

Updated Guest table

Create seating summary for box office sales and integrate that with vip guests seat allocation - 2 points - Completed: A new table "box office_seats" was added to store box office seating summary. On each box office data load, functionality was added to check for unique seat categories and calculate the number of tickets purchased per seating category. Save this information in DB. A new section was created on UI to show a seating summary of box office sales.



Box Office Seating

Category	Total Sold
A	236
B	193

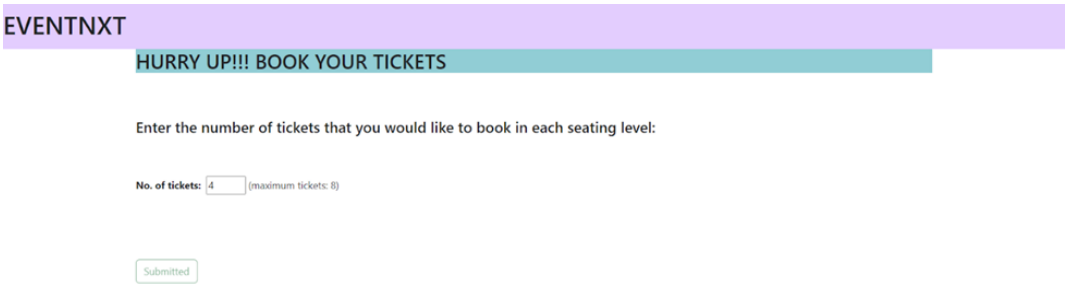
Seating Summary Information

Category	VIP Guests	Total Allotted	Total Committed	Total Sold	Free Seats
A	1	2	0	236	164
B	2	11	0	386	64

Box office and seating summary display

Creating a referral reward email template so employees can refer their friends - 2 points - Completed: The existing referral reward template in the UI was removed. A new email template for referrals was added. Once the guest has accepted the referral the information is saved in DB. The manage referrals section was updated on the UI to show the status of the referral has been accepted or not. The referral acceptance page was updated to provide their email to receive purchase link for the event. The RSVP Invitation template was modified to hide the seating category from the user.

Hiding seating level, max tickets to the sum of all tickets assigned at multiple levels, displaying RSVP data in manage guest table - 2 points - Completed: A Guest Committed Column was created to store the number of seats entered by the guest. The RSVP template was modified so that the seat levels are hidden and the user can now enter a number of seats and click on submit. The maximum number of allotted seats was set as the sum of all the seats in each seating level. A validation for the number of seats entered by the guest was added. The number of seats entered by the guest was stored in the Guest committed column and displayed in the Manage Guest Table.



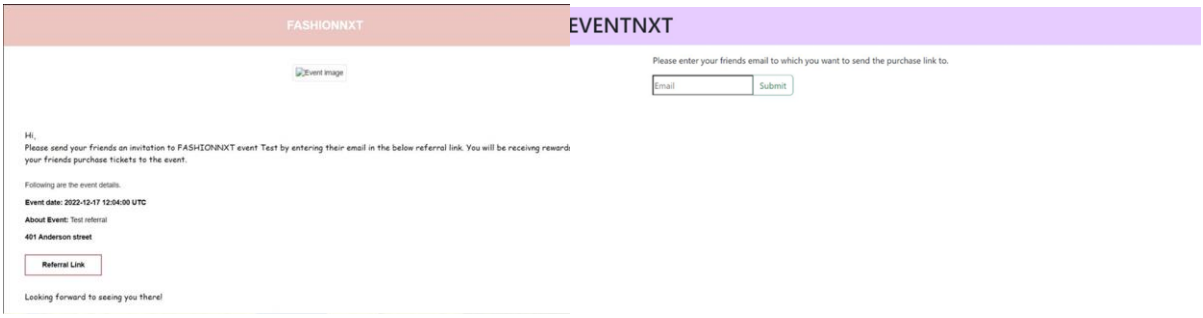
Updated RSVP Screen

Pagination of guests list and box office data - 2 points - Completed: A dropdown (choice) was added for selecting a number of records to be shown at a time. The records are rendered dynamically based on the number chosen. The methods are defined for changing the limit and offset values. These values can be modified based on UI actions. Right, and left arrow buttons were added to call the above-defined methods.

Guest Level Seating Level View Enhancements - Chore (no points) - Completed: The view to show all seating categories and counts was updated. The seating level tabulation was hidden unless the view button is clicked. A "View Seats" button was added to the Manage guests section.

Wrapping the columns of the Manage Guests section in a single row - Chore (no points) - Completed: The Styling of the Manage Guests sections was modified to bring the columns and buttons of the Manage Guests section in a single row.

User Interface for the guest to provide emails to refer people to the event, a new email should be sent to referred guests informing them that they can purchase their tickets to the event and add a Bulk Email feature under Manage Box Office Data to send multiple referral emails- 2 points - Completed: Clicking on the Referral Link opens a new page where the guest will be able to see the details of the event and also enter the email id of the people to refer. On clicking Submit, the data will be populated in the Manage Referrals Section on the UI. An email will be sent to the referred guest with a link where he can buy the tickets for the event. A new checkbox was added to select/unselect all the guest rows in the "Manage Box Office Sales" section. A new button "Bulk Email" was added on the UI that pre-loads all the emails of selected guests, gives the option to select an email template, and then calls bulk email API.



Updated Referral Template Referral Screen

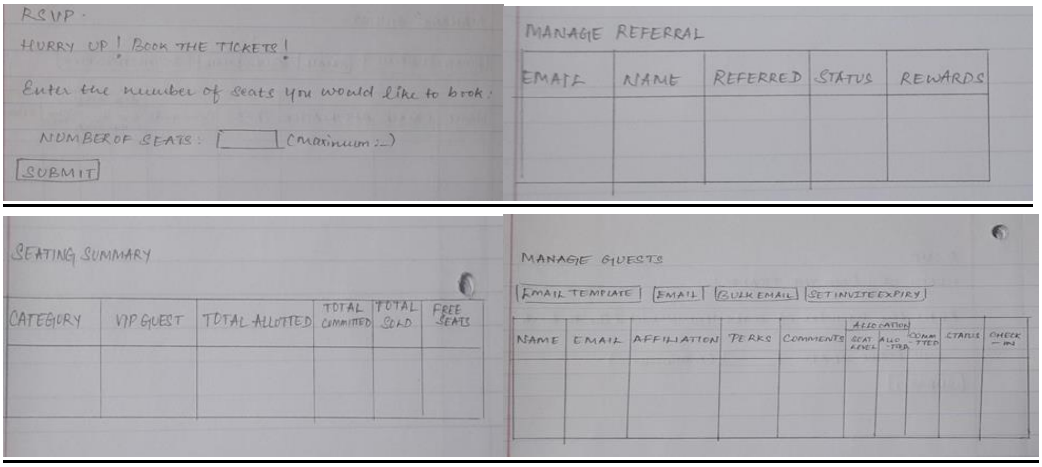
Sending a confirmation mail to the guest once the guest provides RSVP and setting the default FROM mail id in the send email popup to the email address of the organizer who has logged in – 2 points – Completed: Once the guest submits the number of seats in the RSVP a mail will be sent to the guest. The mail provides the confirmation to the guest and contains the number of seats that the guest has requested the organizer to book. Clicking on the send email action will display a popup with the templates available. The FROM mail id is now set as default to the email address of the organizer who has logged in to the application.

Adding a Forgot Password Button in the Login Page - 2 points - Not Started: This functionality enables the user to reset the password

Integrate different statuses to the status column in the guest table and display it on the UI - 1 point - Partially Implemented: This function displays the status of each action such as Guest Added, Guest Invited, RSVP Sent, etc. in the Manage guests table.

Displaying a message on validating the Username and Password in the Login page - 1 point - Not Started: This feature displays a message on the user's screen after validating the correctness of the Username and Password

Lo-fi UI Mockups



Understanding the Legacy Code and Code modifications

We went through the models, views, and controllers in the application to understand the flow of the code. Each of us took a separate section of the application such as Seating, Manage Guests, RSVP, Box office, etc., and understood the working of the same. During our scrum meetings, each of us explains to the team what we went through and showcase our understanding. Once, we had a clear understanding of the flow of the application, we started assigning user stories that included both bug fixes and the incorporation of new features.

The major changes made to the application were making the seating summary section editable so as to edit, update or delete records; Integrate box office data to the application so, the organizer does not have to check the data separately in an excel file and can also perform actions such as Sending an email and deleting the record for those users; Setting up the RSVP page so that the users can mention their interest in attending an event; Referral reward system where the user can send an email to refer a friend. We also added and removed some of the columns in the database which is summarized in the ER diagram below:

Database schema

Team Roles

The team consisted of 6 members: Sravya Gopireddy, Kartheeka Bokka, Aarthi Vasudevan, Srujana Katta Reddy, Akruthi Srikanth, and Aishwarya Vidiyala. The Scrum Master and Product Owner for each iteration were as follows:

	Product Owner	Scrum Master
Iteration 0	Sravya Gopireddy	Srujana Katta
Iteration 1	Aishwarya Vidiyala	Akruthi Srikanth
Iteration 2	Aarthi Vasudevan	Kartheeka Bokka
Iteration 3	Srujana Katta	Sravya Gopireddy
Iteration 4	Akruthi Srikanth	Aishwarya Vidiyala
Iteration 5	Kartheeka Bokka	Aarthi Vasudevan

From the beginning till the end of the project, every team member took a user story and worked on all the components of the story which included both frontend and backend.

Scrum Iterations Summary

Iteration 0: (No points)

In this iteration, we met with our customer to understand what has been implemented in the past and what new features or bug fixes were required to be implemented. We also got feedback on what sections of the application require enhancement in the code and what new features could be added. Among the different sections like Seating, RSVP, Referral, etc. which needed to be prioritized was discussed in order for the user to use the application as early as possible.

Iteration 1: (Chores) - No points

In this iteration, we did research on ticketing portals like Ticket master and Ticket tomato, Event management sites like Eventbrite and Evite, and other popular sites to get an understanding of how these websites are designed to check the feasibility of incorporation of those features into the Eventnxt application. We also set up the Heroku account and deployed the project into Heroku.

Iteration 2: (5 points)

In this iteration, we implemented new features as well as worked on fixing the bugs in the existing code. We added a feature to send a Bulk email to the users so that the organizer can decide to send emails to a single or multiple users. We made the seating arrangement table editable to perform actions like Edit, Update or Delete a record from the table. We integrated the Seating Summary section with the RSVP system to show the correct count of values on the Seating Summary table. We set up the send grid email system and pointed it to the current official email address.

Iteration 3: (8 points)

In this iteration, we added a section called “Load Box office data” which enables the organizer to integrate the box office data with the application. We fixed the RSVP email and RSVP confirmation Email templates as per the suggestions received from the customer. We created a User interface for the guest to provide their RSVP and thereby, store this information in the database to be displayed in the seating summary table. We worked on expiring the RSVP link once the organizer decides to stop accepting RSVPs for a particular event.

Iteration 4: (4 points)

In this iteration, we worked on saving the headers of the integrated box office data so that the user does not have to change the headers every time they click on Load Box Office Data. We added new columns, Perks, and Comments to the database and to the UI for the organizer to assign perks or add additional comments for any particular VIP guest. We also made all the

columns except for the Email column in the manage guests section editable. We modified the manage guests section to show a seating allotment summary for each guest.

Iteration 5: (8 points)

In this iteration, we created a section to show the Seating summary for Box office Seating and in turn integrated it with the Seating allocation of the VIP guests. We also worked on hiding the seating summary information in the Manage Guests section and made it visible only when the user clicks on the View Seats button. We worked on hiding the seat level information on the RSVP page and also in the RSVP email. We added a new column, Guests committed to the Manage Guests section to know the number of users who have accepted the RSVP. We created an RSVP confirmation email that sends the user, the number of seats that were confirmed after submitting an RSVP. We also added a pagination feature to the VIP guest list and the box office data list for easy navigation between the pages.

Other Implemented User Stories

We created a UI for the guest to provide email ids to refer people to the event, and an email was sent to the referred guests that gives the purchase link to buy the tickets. We also added a Bulk Email feature under Manage Box Office Data to send multiple referral emails. We created a confirmation mail to the guest once the guest provides an RSVP and set the default FROM mail id in the send email popup to the email address of the organizer who has logged in.

Customer Meeting Dates and Description

SNo	Date & Time	Current meeting discussions	Action Items for next meeting
1	09/21/22 5pm	Discussed an overview of the legacy application and main areas expected to be improvised.	To deploy the current application on the local machine and do research on event websites, ticket sales, RSVP, and referral links from other widely used ticket booking websites.
2	09/28/22 5pm	Demonstrated the application running from local set up and discussed the research on third-party event applications and ticket websites.	To set up the rails credentials and get the application up on Heroku.

3	10/12/22 5pm	Showcased the application deployed on Heroku and explained the analysis on referral options.	To get the send grid API part to work for sending emails to guests depending on event status and make the seating level editable.
4	10/19/22 5pm	Demonstrated the send grid API to email guests and discussed the fields on the seating summary page and identified certain flaws from the past team application.	To enhance the email sending part by adding a bulk email template and fixing discrepancies on the seating summary table.
5	10/26/22 5pm	Illustrated the bulk email part to send emails to multiple guests at once, configurable seating summary page to create, destroy and update seats, and the fixes done on the seating summary page for the mismatching columns.	To incorporate the feature to add box-office data and display it on the front-end, resolve the issues with the existing RSVP email template,
6	11/02/22 5pm	Showcased the box-office sales upload by uploading it either during creating an event or updating it and the data display on the UI. Discussed the progress of fixing the RSVP invitation template.	To add a feature to set up auto expiration for RSVP links based on timestamp and fix the RSVP front-end screen to save the user response on the back-end.
7	11/09/22 5pm	Demonstrated the RSVP link expiration, fix on RSVP email invitation and confirmation templates, and reserved user response.	To add individual seating summary display on the Manage guests section, save the headers mappings for box office upload, and utilize those when uploading new files.
8	11/16/22 5pm	Showcased the progress on the box office part and displayed the different seating levels for each guest.	To add extra columns - Perks and Comments at the guest level and make all the fields editable except for email. To add a feature to create box office sales data and integrate the same with VIP information.

9	11/23/22 5pm	Illustrated the seating summary display at the guest level, configurable guest data for all the fields excluding email, additional columns included and header mapping saved when uploading box-office data.	To add pagination to the manage guests section to handle bulk records and hide the individual seating split up for guests and instead display an extra column on the manage guest page for total committed seats.
10	11/30/22 5pm	Demonstrated the pagination completed on the manage guests section and feature deployed to display the box office sales data merged with the VIP guest details. Also showcased the feature added to hide the seat split up to guests and an additional column for total committed seats on manage guests.	To get the referral email part working and add refine the referral email template.
11	12/07/22 5pm	Demonstrated the Referral part working and got feedback for the overall progress done by our team on the application.	To add the necessary documentation for references and details on sections to be enhanced in the future.

Behavioral-driven development (BDD)

In accordance with the BDD process, we have followed an agile method of software development. We initially discerned the significant expectations of the application, and desired outcomes and did research from various perspectives to fix the flaws in the existing application and add features. We also created Lo-fi mockups to empathize with the users and plan for refining the current front-end models. We had regular scrum calls thrice a week, to create user stories, discuss each teammate's progress, stay on track and figure out the major roadblocks. We also had weekly meetings with the client to prioritize our tasks and work accordingly. We showcased our recent updates, received feedback, and incorporated the suggestions in the following connects. Hence through proper communication, we merged both business and technical insights into our application to meet the requirements.

Test-driven development (TDD)

And applying the TDD process, we implemented certain phases that involve creating unit test cases even before developing the real module. We first created unit test cases to evaluate the main flow of the application and ensure the feature expectations. Based on test failures, we kept modifying the modules to rectify the flaws, re-tested the new code, and deployed it

successfully meeting the planned outcomes. We generated good test coverage reports every iteration using RSpec and Cucumber frameworks.

On the whole, BDD and TDD helped us build a resilient software product satisfying the business needs, develop a cost-efficient model that withstands in the long run, and provides a valuable service. Following is the coverage report obtained using RSpec:

Search:

File	% covered	Lines	Relevant Lines	Lines covered	Lines missed	Avg. Hits / Line
app/helpers/application_helper.rb	100.00 %	2	1	1	0	1.00
app/helpers/events_helper.rb	100.00 %	2	1	1	0	1.00
app/helpers/guests_helper.rb	100.00 %	2	1	1	0	1.00
app/helpers/welcome_helper.rb	100.00 %	2	1	1	0	1.00
app/models/application_record.rb	100.00 %	3	2	2	0	1.00
app/models/user.rb	75.00 %	26	12	9	3	0.75

Configuration Management Approach

Applying the configuration management approach, we worked on building a reliable and organized model using configuration management tools. First, Git served as a significant codebase for managing configuration data. We were able to have version control on the Git repository. Every time we do a change, we first merged out from the master branch, deployed the changes in the new branch, tested the same, and then merged the code to the master branch by creating pull requests. This approach helped us to view the changes done and the files modified, have code reviews, organize our tasks and understand the complexity of each user story. Additionally, the process saved our application from unpredicted code failures. This provided the facility to roll back or revert the changes if something went wrong. We had 17 feature branches and 5 releases.

We also used Docker as a configuration tool for containerizing our application based on the configuration file. The docker process results in a snapshot of the application that is then committed to the GitHub repository and deployed on Heroku. Thus, configuration management aided in having better visibility of the application, handling the activities, and addressing the different business requirements.

Heroku Production Release Process

We used Heroku as the cloud platform technology for deploying our code changes to production. We ensured thorough testing of the code by adding test cases, testing the application on the local host, creating a feature branch, and pushing the code to the feature branch. Once, the feature branch was deployed and tested on Heroku, we merged the changes from the feature to the master branch and again tested the deployment of the master branch on Heroku. Some of the issues faced while deploying the code to Heroku were: During the initial deployment, we faced a “production server failed to load” error for which we removed the "config.assets.compile = false" property to fix the issue. Also, since, the free tier for the Heroku cloud platform ended, we faced an application error as a result of which we couldn’t view our

application in production. Hence, we upgraded our Heroku version to view the deployed application.

Issues with AWS, Cloud9, GitHub, etc.

We used the GitHub repository to have a code base for the application and to push the code to Heroku by directly connecting to our GitHub repository. Also, GitHub was useful when we had to perform code reviews in which one can share their pull request so that the other developers can review the code changes before they are merged to the master branch. At times, we faced merge conflict issues while merging the changes to the master branch. We used Heroku for deploying our application.

Code Climate/Simple Cov Benefits

We used code climate to test the quality of our code. It allowed tracking the progress and issues in the application in a timely manner with the help of which we were able to reduce code flaws during the development stage as well during the deployment in the production. We also used Simplecov gem that uses the RSpec command to help us understand how much percentage of our was tested. This helped us to determine which sections of the application required more testing and which sections were thoroughly tested.

Cucumber and RSpec

We have added all the test cases created in cucumber and RSpec have added to the public GitHub repository.

Repository Contents

The GitHub Repository contains all the models, databases, controllers, views, JavaScript files, helper files, readme file that contains the docker commands, documentation that covers iteration and final reports, and other files like Gemfile and Dockerfile. The repository also contains a sample tickets.xlsx file for testing with the box office data.

Links

Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2597192

GitHub repository: https://github.com/tamu-edu-students/EventNXT-606

Heroku Deployment: https://eventnxtdev.herokuapp.com/

Presentation (.pptx file): https://github.com/tamu-edu-students/EventNXT-606/blob/master/documentation/Fall2022/EventNXT-Poster.pptx

Demo Video: https://youtu.be/k9GVwUJeJCg