CSCE 606 Final Progress Report

Fall 2023

Team Members and Roles

- Huy Lai : Product Owner
- Shreya Gubbi Prakash : Scrum Master Iteration 0 - 3
- Neha Manghnani : Scrum Master Iteration 4 - 5
- Thomas Manzini : Developer

Links

Heroku Deployment
Slack
Pivotal Tracker
GitHub
Presentation and Demonstration Video

Project Summary

The Contract Management System (CMS) developed for the Brazos Valley Council of Governments (BVCOG) during the Spring 2023 semester by the CSCE 606 class has undergone significant enhancements and adaptations in the Fall 2023 iteration. These updates vary in type and technical requirements and largely fall into two categories. The first category of changes are simple updates to the user interface. The second category of changes requested by the customer is to change the functionality of existing contract entry forms. This iteration emphasized a refined user access hierarchy, granting distinct privileges to Admins, Gatekeepers, and Users, with the latter now empowered to initiate contract creation. User interface enhancements, including restructured contract entry fields and streamlined data inputs, aimed to augment user experience and ensure precise data capture. Stakeholders, including BVCOG administrators, Gatekeepers, and Users, collectively engaged in ensuring the system's alignment with organizational needs, reinforcing administrative oversight, and bolstering data accuracy.Their goal was to grant more control to leaders, improve user-friendliness, and guarantee accurate information. This updated system aimed to simplify contract management for BVCOG, prioritizing ease of use, empowering leaders, and ensuring precise details, addressing the initial project challenges.

The current implementation of the project supports all of the requirements. One huge factor of the completion of the project was that their top three stakeholders provided us with a requirements document at the beginning and we had in-person meetings in which we showcased new features with immediate feedback. We also had weekly updates via mail. The platform is deployed to Heroku and the client has already started testing it.

User Stories

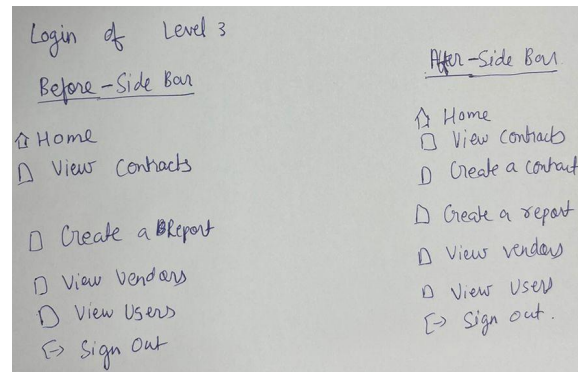Structure: User Story # | Implementation Status | Points
#1 | Implemented | 2
Feature: Users Can Create Contracts
As a platform user with "Level 1" access
So that I can create a new contract for approval
I want to add a new contract entry to the database.

#2 | Implemented | 2
Feature: Revoke Contract Creation Privilege from User Level 2
As a platform Gatekeeper user
So that I cannot create contracts that I then can approve without review
I want to remove the ability to create contracts.

#3 | Implemented | 1
Feature: Change User Access Level Description
As a platform user (of any level)
So that I can understand user access levels easier
I want to rename User Access Levels to more human-friendly descriptions

LoFi Mockup

Actual User Interface

View Users

👥 Users				Search by first or last name
First Name	**Last Name**	**Level**	**Program**	**Program Manager**
Tyler	Brakus	Admin	Program 4	No
Lawrence	Williamson	User	Program 4	No
Anika	Ritchie	User	Program 5	No
Veola	Hayes	Gatekeeper	Program 1	No
Beth	Weissnat	Admin	Program 2	No
Giuseppe	Schowalter	Gatekeeper	Program 5	No
Annamae	Sawayn	Gatekeeper	Program 1	No
Melvina	Huels	Gatekeeper	Program 2	No
Stefan	West	Admin	Program 3	No
Nicky	Kris	Gatekeeper	Program 3	No

1 2 3 4 5 6 Last» Next ›

#4 | Implemented | 2
Feature: Change the title displayed from "Key words" to "Search key words"
As a platform user (of any level)
So that I can understand how to search for a contract by keywords
I want to alter the search box text to make this more clear

#5 | Implemented | 2
Feature: Change Vendor dropdown menu to a text entry
As a platform user (of any level)
So I can search through Vendors more efficiently
I want to be able to search Vendors by name.

#6 | Implemented | 4
Feature: Vendor Text Entry Should Display Applicable Vendors
As a platform user (of any level),
So I can search through Vendors more efficiently,
I want the Vender Text Entry to predict the Vendors' name
And give all the Vendors who fit what I put in the text.

For this user story, we have used the JQuery autocomplete feature, which filters the list of
vendors based upon the text entry on the text field.

#7 | Implemented | 2

Feature: Move Fields for the Create Contract Screen around

As a platform user (of any level)

So that I can create a contract more efficiently,

I want the field entry to be in an order which is logically placed, starting with basic information.

New Order Requested by Client

Title	Contract Duration Fields
Number	Contract Value Fields
Vendor	Contract Documents
Description	
Contract Type	
Point of Contract	
Program	
Entity	
Key Words	

LoFi Mockup

CONTRACT INFORMATION
Title
Number
Vendor
Description
Contract type
Point of contact
Program
Entity
Key words→ (Search Key word)

Start date   End date
Dollar Amount   Amount duration
End trigger
Initial term amt   Initial term duration   Remaing Renewals
Requires rebid ☐
Contract documents
⬆ Choose Files

#8 | Implemented | 1
Feature: Expiry Reminder
As a Gatekeeper User who is reviewing a Contract
I want to remove the ability to send a expiry reminder
So that this ability no longer exists.

#9 | Implemented | 1
Feature: Invitation Email
As an Administrator
I want to change the second paragraph of the invitational email
So that invited people better understand how to accept this invitation.

#10 | Implemented | 2
Feature: Change "View Vendors" to "Vendor Rating"
As a platform user (of any level)
I want to change "View Vendors" menu to "Vendor Rating"
So that I can understand the vendors better.

#11 | Implemented | 3
Feature: Field Hints
As a platform user who can create contracts,
So that I can understand what information to enter in the fields while filling the contract information.
I want hints to be displayed when I hover my mouse over each field.

#12 | Implemented | 2
Feature: End Trigger
As a user who can create contracts
I want to specify that certain contracts have no end date
So that I can allow the length of the contract to go unspecified.

#13 | Implemented | 2
Feature: Final End Date
As a platform user who has created a Limited Term Contract with a specified number of renewals
I want to specify the final end date
So that other users know when a renewed contract will finally expire.

#14 | Implemented | 6
Feature: Renewal Options
As a platform user who has created a Limited Term Contract
I want to specify the number of renews that a contract can have
So that contracts with optional renewals are included in the system.

#15 | Implemented | 2
Feature: Current End Date
As a platform user who can create contracts
I want to be able to specify that certain contracts have a determined end date
So that the length of contracts can be determined.

#16 | Implemented | 1
Feature: View Only Applicable Contracts
As a Gatekeeper User,
I want to view contracts that are a part of the same entity as I am,
So that I do not see every contract within the system.

#17 | Implemented | 6
Feature: Reject a Contract
As a Gatekeeper user,
I want to be able to reject contracts
So that contracts with inaccurate information do not populate the database.

As an addition to this feature, the contract decision history was developed which logs the status of the contract on every update made and displays the list of contract statuses over a period of time along with the rejection reason (if the contract was rejected).

The gatekeeper rejects the contract with a rejection reason

The contract logs are maintained in 'Contract Decision History' where the rejected event and the reason for rejection are noted for the user to review and update the contract accordingly.

☰ Contract Details	
Title:	Example Contract Report 12
Number:	w32145
Contract Type:	ILA
Initial Term:	2 years
End Trigger:	Continuous
Point of Contact:	Alan Becker
Vendor:	Vendor 10
Entity:	Entity 4
Program:	Program 3
Start Date:	December 01, 2023
Requires Rebid:	No
Amount:	$20.00 per month
Total Amount:	Not Applicable
Status:	Rejected

Set to "In Progress"

ⓘ Description

Key Words:

Description: ed

📄 Contract Documents

File Name   Document Type

Contract Decision History

Event: Created
User: Example User
Date: 2023-12-01 09:22:49 UTC

Event: Moved to "In Progress"
User: Example User
Date: 2023-12-01 09:22:49 UTC

Event: Moved to "In Review"
User: Example User
Date: 2023-12-01 09:29:06 UTC

Event: Rejected
User: Gatekeeper User
Date: 2023-12-01 09:32:03 UTC
Reason: This contract is rejected due to invalid dollar amount value. requesting for a review on the same.

#18 | Implemented | 1
Feature: Add Items to "Contract Type"
As a platform user who can create a contract,
So that I can specify more contract types
I want more contract types.

19 | Implemented | 4
Feature: Contract Value Fields (Total Value)
As a platform use who can create contracts
I want to specify the total value of certain contracts
So that contracts with monetary value are included within the database.

#20 | Implemented | 1
Feature: Contract Value Fields (Not Applicable)
As a platform use who can create contracts
I want to specify value type as "Not Applicable"
So that contracts with no monetary value are included within the database with total amount equal to $0

#21 | Implemented | 4
Feature: Contract Value Fields (Calculated Value)
As a platform user who can create Contracts,
I want to be able to specify that certain contracts have no monetary value
So that contracts that have no monetary value can be accounted for within the database.

The Contract Value field is the total value in dollars of the contract. A contract can have the value field as Not Applicable (a contract which does not have a monetary value associated with it), Total value (the overall contract value regardless of the value mentioned interim amount) and Calculated value (value of the contract based calculated by the value of the contract for a duration times the time period for which the contract is valid).
We have a select field to select the 'Value Type' required to calculate the Total amount value of the contract. It is a dynamic view which shows an additional field to enter the total amount if the value type 'Total Value' is selected (to get the user to input the total amount of the contract). If the selected value type is 'Calculated Value' , we have a logic implemented to get the total value based upon the contract value for a duration times the duration of the contract. If the selected value type is 'Not Applicable' , the total contract value = $0. For 'Calculated Value' and 'Not Applicable' value type the user input for total amount is hidden to avoid user input error in the contract.

Value Type

    Total Value

Total Value

    80000

Interim Amount

$ 20   per   month

Initial Term

🕐 2   Initial term duration   year

Value Type

Calculated Value

☰ Contract Details	
Title:	Example Contract Report 12
Number:	w32145
Contract Type:	ILA
Initial Term:	2 years
End Trigger:	Continuous
Point of Contact:	Alan Becker
Vendor:	Vendor 10
Entity:	Entity 4
Program:	Program 3
Start Date:	December 01, 2023
Requires Rebid:	No
Amount:	$20.00 per month
Total Amount:	$480.00
Status:	In Progress

Value Type

Not Applicable

Program:	Program 3
Start Date:	December 01, 2023
Requires Rebid:	No
Amount:	$20.00 per month
Total Amount:	Not Applicable
Status:	In Progress

Shows 'Not Applicable' if the Total amount is $0 (User story #27 mentioned below).

#22 | Implemented | 1
Feature: Sorting Contracts by Type
As a platform user (of any level),
I want to be able to sort the contracts by type,
So that I can parse this information more easily.

#23 | Implemented | 1
Feature: Generate Contract Reports by Contract Type
As a platform user,
I want to be able to create a report that contains contract of a specified type,
So that I can parse this type of information more easily.

#24 | Implemented | 1
Feature: Remove Upon Completion
As a platform user who can create contracts
I want to remove the "Upon Completion" end trigger
So that these types of contracts do not exist.

#25 | Implemented | 3
Feature: Remove Renewal Fields
As a platform user who can create contracts
I want to remove the renewal fields
So that contracts have consistent and accurate wording

#26 | Implemented | 1
Feature: Remove Required Renewal Button
As a platform user who can create contracts
I want the "Requires Renewal" button
So that contract renewals can be handled in a different method

#27 | Implemented | 1
Feature: **Show "Not Applicable" when viewing Contracts with $0 value**
As a platform user for any level,
When the contract total amount is $0 I want to show it as "Not Applicable"
So that I can understand that this contract does not have a value.

#28 | Implemented | 1
Feature: Remove Maximum Extensions
As a platform user who can create contracts
I want to remove the maximum number of extensions
So that contracts with these extensions can be handled differently

#29 | Implemented | 1
Feature: Change "Remaining Extensions" to "Number of Extensions"
As a platform user who can create contracts
I want to change "Remaining Extensions" to "Number of "Extensions"
So that Extension behavior is more consistent.

Scrum Iteration Summaries

Iteration 0

Build LoFi mockups for the user interface changes.
Set up meetings with the clients. Decided on team roles.
Gain access to Pivotal Tracker and GitHub
Points: 0

Iteration 1

Worked on modifying contract creation privileges.
Start modifying the User Interface per client requirements.
Deploy the project to Heroku.
Points: 6

Name	Points
Huy Lai	2
Shreya Gubbi Prakash	2
Thomas Manzini	1
Neha Manghnani	1

Iteration 2

Change Vendor Dropdown
More user interface changes
Trivial email sending changes
Points: 12

Name	Points
Huy Lai	2
Shreya Gubbi Prakash	4
Thomas Manzini	0
Neha Manghnani	6

Iteration 3

Debugging the vendor text entry.

Points: 2

Name	Points
Huy Lai	0
Shreya Gubbi Prakash	2
Thomas Manzini	0
Neha Manghnani	0

Iteration 4

Change contract duration fields per client request.

Add field hints

Points: 15

Huy Lai	6
Shreya Gubbi Prakash	0
Thomas Manzini	8
Neha Manghnani	0

Iteration 5

Add ability to reject contracts
Polishing the user interface
Add Quality of Life features
Points: 18

Huy Lai	5
Shreya Gubbi Prakash	4
Thomas Manzini	3
Neha Manghnani	5

Iteration 6[1]

Polish the user interface
Ensure testing is sufficient
Documentation
Points: 8

Huy Lai	0
Shreya Gubbi Prakash	3
Thomas Manzini	2
Neha Manghnani	3

---

[1] Not a real "iteration" in terms of implementing new features.

Client Meeting Dates

At the direction of the customer, we will be providing weekly update emails detailing the work that has been done since the last update. During our initial meeting, the customer indicated that meeting on a weekly basis would not be productive and that we should instead meet at 3 week intervals to answer questions and provide in person updates.
We received additional requirements from the client which were communicated by mail.

1. Customer Meeting #1 2023-09-12 11:00-14:30 CDT
   The customers told us the exact requirements, we also got to clarify all our doubts and know the exact want of the customer.
2. Customer Meeting #2 2023-10-06 13:00-15:00 CDT
   We gave a demo of all the UI changes needed for example -Changing the title displayed from "Key words" to "Search key words" and many such changes, we also gave a demo on the ability to create contracts to User Level 3, remove the ability to create contracts from User Level 2, change User Access Level Descriptions for User Levels 1, 2, & 3.
   We received the texts to be added for the field hints from the client.
3. Customer Meeting #3 2023-11-01 11:30-13:00 CDT
   We gave a demo of the Vendor Dropdown changed to Text field and Menu Item Change, clarified requirements related to Contract Duration and Value fields to added and dynamic updates required for it. In this meeting, minor changes were requested by the client on the create contract and create report features.
4. Customer Meeting #4 2023-11-20 15:00-17:00 CST
   We gave a demo for the completed user stories, some of the major ones including Rejecting Contract which included maintaining a decision history for the contract containing the rejection reason, explained how the Contract Value Fields are implemented and how it works. The client requested for UI updates and text changes. During this meeting we asked for feedback and review on the work performed by the team and received a positive feedback from the client

Repository Summary

We included a README.md into the repository that specifies the prerequisites and how to install and deploy the application.

BDD/TDD Process Summary

While we tried to follow a Test-Driven-Development process, we struggled early on due to a lack of experience with rails, and especially Rspec and cucumber. Testing early on

revolved around fixing many of the cucumber tests, essentially rewriting many of the tests involving regular expressions. As the project progressed, this testing became more fully featured. We mainly focused on writing tests that matched the specifications originally set by our clients. Consequently, during our meetings with them, we improved on specific features. As the project progressed, more focus was put onto Rspec testing, which had been behind. Due to changed expectations, many of the previously written tests would fail and we would have to rewrite those tests to fit the new requirements from the client. For example, due to the changed behavior of the vendor dropdown, many of the contract creation tests would break. Additionally, changing the contract duration and value fields would introduce failures not previously seen in prior tests.

Configuration Management

At the time of writing: 2 Branches, and 100 closed Pull Requests. No spikes were required. We had 2 main branches; test, where we tested new features, and main, which we deployed into production.

Production Challenges

Due to the documentation from the previous team, getting deployed was a relatively smooth experience. However, the rails master key required to deploy the application was not sufficiently explained well enough. We hope that the importance of this key is well evident within the documentation.

Challenges With Tooling

The previous team decided to work with GitHub and set up their machines to have the same Ruby version from day 1. Most of the issues we faced were mostly merge conflicts, which is common when each member is working on a different branch. Thus, each pull request before merge needed approval and the most critical one was merged first to the test branch. Furthermore, we utilize GitHub Actions which were an extra evaluative step preventing us from merging a pull request if any test from cucumber was failing.

Other Tooling

The Spring 2023 team used the following tools
- [Devise](): For user authentication and password resetting
- [Devise_inviteable](): For user invites
- [Kaminari](): Table pagination
- [Bulma_rails](): Styling

- [Polar OSO](#): Role-based Access Control
- [Whenever](#): Automated cron jobs

The Fall 2023 team added the following tools
- [JQuery](#): autocomplete the vendor text entry