Project Report

All Pay Auction
CSCE 606 - Spring 2025

April 26, 2025

Contents

1 Project Summary

The AllPayAuction application addresses the need for a platform where sellers can conduct all-pay auctions. In this model, all bidders pay their bid amounts, but only the highest bidder wins the item. The application facilitates this by providing separate interfaces for sellers and buyers. Sellers can create auctions with specific parameters (description, duration, minimum bid, increment), while buyers can place bids and track auction status. Key features include user authentication (initially basic, later Google OAuth), an auction block displaying item information, and basic account balance management. The implementation includes functionality for automated bid settlement, seller-initiated auction finalization, and handling multiple bids. UI improvements, JavaScript refactoring, and test automation were also significant development focuses.

Contrasting this with the initial project description, the implemented application stays largely true to the core requirements. Notable additions and refinements include enhancements to user authentication (Google OAuth), more robust UI development, and a focus on code quality and testing. The project also delivered features such as uploading multiple images, email verification, password recovery, and currency formatting which were not explicitly mentioned in the original description. The stakeholders for this application are primarily the sellers who wish to auction items and the buyers who participate in these auctions. The developers and project managers are secondary stakeholders who ensure the application meets the needs of the primary stakeholders.

2 User Stories

2.1 Sprint 1

1. Feature: Landing Page (0.5 point)
   As a user,
   I want to create an account as either a buyer or seller
   so that I can interact with the auctions
   Status: Completed.

2. Feature: Buyer's Login Page (1 point)
   As a buyer,
   I want to log in,
   so that I can participate in auctions.
   Status: Completed.

3. Feature: Seller's Login Page (1 point)
   As a seller,
   I want to log in,
   so that I can set the auctions.
   Status: Completed.

4. Feature: Buyer's Account Creation Page (1 point)
   As a new buyer,
   I want to create a buyers account,
   so that I can login and view auction items to purchase.
   Status: Completed.

5. Feature: Seller's Account Creation Page (1 point)
   As a new seller,
   I want to create an seller's account,
   so that I can sell items.
   Status: Completed.

6. Feature: Database Creation (1.5 point)
   As a platform administrator,
   I want to create and maintain a reliable database that securely stores buyer and seller information
   so that users can safely register, log in, and manage their accounts.
   Status: Completed.

2.2 Sprint 2

1. Feature: Seller Settles the Auction (1 point)
   As a seller,
   I want to settle the auction after the closing date,
   so that I can determine the winner and collect payments from all bidders.
   Status: Move to sprint 3.

2. Feature: User delete account (1 point)
   As a user,
   I want to be able to delete my account
   So that I can remove all my personal information from the system
   Status: Move to sprint 3.

3. Feature: User update profile information (1 point)
   As a user,
   I want to be able to update my profile information

So that I can keep my information up to date.
Status: Move to sprint 3.

4. Feature: Database Organization (2.5 point)
   As an admin,
   I want a database ,
   So that I can track my ongoing auctions and create new ones easily.
   Status: Completed.

5. Feature: Seller Creates an Auction (1 point)
   As a seller,
   I want to create an auction with a closing date,
   So that I can start accepting bids for my item and ensure the auction ends at a specific time.
   Status: Completed.

6. Feature: User Places a Bid (1 point)
   As a user,
   I want to place a bid in the auction,
   so that I have a chance to win the item.
   Status: Completed.

7. Feature: View auction product item (1 point)
   As a buyer,
   I want to view detailed information about an auction item,
   So that I can make an informed decision before placing a bid.
   Status: Completed.

8. Feature: Seller Settles the Auction (1 point)
   As a seller,
   I want to settle the auction after the closing date,
   so that I can determine the winner and collect payments from all bidders.
   Status: Completed.

2.3 Sprint 3

1. Feature: User delete account (1 point)
   As a user,
   I want to be able to delete my account
   So that I can remove all my personal information from the system.
   Status: Completed.

2. Feature: User update profile information (1 point)
   As a user,
   I want to be able to update my profile information
   So that I can keep my information up to date.
   Status: Completed.

3. Feature: UI enhancement (2 point)
   As a buyer,
   I want a dedicated interface to navigate through the website,
   So that I can track the ongoing auctions and participate in one easily.
   Status: Completed.

4. Feature: Adding Funds to account (3 point)
   As a buyer,
   I want to add funds to my account,
   So that I can participate in the auction.
   Status: Completed.

5. Differentiate liquid and asset balances in seller (0.5 point):
   As a seller,
   I want to see my liquid and asset balances displayed separately,
   so that I can easily understand my available funds and the value of my held assets.
   Status: Completed.

6. Differentiate liquid and asset balances in buyer (0.5 point):
   As a buyer,
   I want to see my liquid and asset balances displayed separately,
   so that I can easily understand my available funds and the value of the items I've won.
   Status: Completed.

7. Add innate value of AuctionItem from Seller (0.5 point):
   As a seller,
   I want to be able to specify the innate value of an AuctionItem when creating it,
   so that buyers have a baseline understanding of the item's worth.
   Status: Completed.

8. Add minimum increment in auction item (0.5 point):
   As a seller,
   I want to be able to set a minimum bid increment for an AuctionItem,
   so that I can control the pace and progression of bidding.
   Status: Completed.

9. Add isArchived flag in AuctionItem (0.5 point):
   As an administrator or seller,
   I want to be able to archive an AuctionItem by setting an "isArchived" flag,
   so that I can remove it from active listings without permanently deleting it.
   Status: Completed.

10. Feature: Seller Settles the Auction (1 point)
    As a seller,
    I want to settle the auction after the closing date,
    so that I can determine the winner and collect payments from all bidders.
    Status: Move to sprint 4.

2.4 Sprint 4

1. Feature: Bid settlement (2 point)
   As a system, I want to automatically settle all bids when an auction ends, so that all participating bidders are charged, regardless of winning or losing.
   Status: Completed.

2. Feature: Seller Settles the Auction (3 point)
   As a seller, I want to be able to manually finalize an auction and trigger bid settlement, so that I can control the auction's end and ensure all payments are processed.
   Status: Completed.

3. Feature: Profile Settings UI Improvement (1 point)
   As a user, I want an improved and intuitive profile settings interface, so that I can easily update my personal information and preferences.
   Status: Completed.

4. Feature: Add Cucumber to GitHub Actions (1 point)
   As a developer, I want to integrate Cucumber tests into our GitHub Actions workflow, so that we can automate behavior-driven testing and ensure consistent application functionality with each commit.
   Status: Cancelled. Did not have time to implement in the end.

5. Feature: Syncing Bids for multiple buyers (1 point)
   As a buyer, I want to see real-time bid updates from other buyers, so that I have an accurate and up-to-date view of the auction's progress.
   Status: Completed.

6. javascript refactoring (1 point):
   As a developer, I want to refactor the JavaScript code to improve its maintainability, readability, and performance, so that the application is more robust and scalable.
   Status: Completed.

7. implement pagination for various feature (2 point):
   As a user, I want to see paginated results for auction listings, bid histories, and other relevant features, so that I can easily navigate and manage large datasets.
   Status: Completed.

8. Update ”place bid” to consider min increment (1 point):
   As a buyer, I want the ”place bid” functionality to enforce the minimum bid increment, so that I am prevented from placing bids that do not meet the auction's requirements.
   Status: Completed.

9. Add minimum increment in auction item (1 point):
   As a seller, I want to set a minimum bid increment for my auction items, so that I can control the pace and progression of the bidding.
   Status: Completed.

10. Adding funds to account (3 point):
    As a buyer, I want to be able to add funds to my account via various payment methods, so that I can participate in auctions.
    Status: Completed.

11. Feature: Buyer Dashboard UI Improvement (1 point)
    As a buyer, I want an improved and intuitive buyer dashboard, so that I can easily track my bids, view auction statuses, and manage my account.
    Status: Completed.

3 Team Roles

Team roles			
Sprint	**Product Owner**	**Scrum master**	**Developers**
1	Jason Le	Praewa Pitiphat	Arkan Abuyazid Steve Wang Ruvail Shahzad
2	Praewa Pitiphat	Jason Le	Arkan Abuyazid Steve Wang Ruvail Shahzad
3	Arkan Abuyazid	Steve Wang	Praewa Pitiphat Jason Le Ruvail Shahzad
4	Steve Wang	Arkan Abuyazid	Praewa Pitiphat Jason Le Ruvail Shahzad

4 Accomplishment

4.1 Sprint 1

Sprint 1 Tasks			
Tasks	**Points**	**Assignees**	**Total Points**
- Create Database using PSQL	1.5	Arkan	1.5
- Deploy and host application to Heroku	1	Praewa	1
- Create Landing page - Create Seller's account - Create Buyer's Account	0.5 1 1	Jason	2.5
- Set up Code Climate for GitHub - User's Login page - Seller's Login page	1 1 1	Steve	3

Summary: This sprint saw the successful implementation of eight key user stories, laying a solid foundation for the application. These included core account functionality with the creation of buyer and seller accounts, along with corresponding login pages. The database was successfully created using PostgreSQL, and a landing page was developed to guide users through the account creation process. Supporting the development process, Code Climate was integrated with the GitHub repository for automated code quality analysis. Finally, the application was deployed and hosted on Heroku, providing easy access for testing and review by stakeholders.

4.2 Sprint 2

Sprint 2 Tasks			
Tasks	**Points**	**Assignees**	**Total Points**
- Bid Listing - Bid History - Seller Interface - Seller Creates an Auction - View Auction product item - CRUD action item and Onboarding design (UI)	1 0.5 1 1 1 2	Jason	6.5
- User places a bid - View Auction product item	2 1	Steve	3
- Create DB for auction items - Designing a new database schema	3 -	Arkan	3
- Setup cucumber and Rspec inside docker - Fixing capybara bugs (testing enviroment issue) - Organized all documents - Writing cucumber and Rspec tests for Sprint 1 functions	0.5 2 - 2	Praewa	4.5

Summary: This sprint saw the successful completion of several core functionalities, including the implementation of bid listings, a seller interface, and the ability for sellers to create auctions. Additionally, the application UI was updated. Users can now effectively place bids and view auction product items. However, some user-related features, such as account deletion, profile updates, and the seller's ability to settle auctions, were deferred to Sprint 3. Progress was also made on BDD/TDD testing, which remains ongoing.

4.3 Sprint 3

Sprint Tasks			
Tasks	**Points**	**Assignees**	**Total Points**
- Upload multiple images - Auction Status Visibility - Missing buyer dashboard routing - Google OAuth Implementation - Improve password security - Email Verification - Password recovery/forgot password - Auction Image Display - Homepage Navigation - Default Auction Dates - Currency Formatting	2 0.5 0.25 2 0.5 1 1 0.25 0.25 0.5 0.5	Jason	8.75
- UI Improvement	4	Steve	4
- Add isArchived flag in AuctionItem - Add minimum increment in auction item - Differentiate liquid and asset balances in seller - Differentiate liquid and asset balances in buyer - Add innate value of AuctionItem from Seller	1 1 1 1 1	Arkan	5
- User delete account - User update profile information - Organized all documents	2.5 2.5 -	Praewa	5

Summary: This sprint saw the successful implementation of several key functionalities, significantly enhancing the platform's user experience and security. Notable achievements include the ability to upload multiple images for auction items, clear visibility of auction statuses, and improved routing for buyer dashboards. Security was bolstered through Google OAuth implementation, enhanced password security, email verification, and password recovery features. User experience improvements encompassed streamlined auction image display, intuitive homepage navigation, default auction date settings, and consistent currency formatting. Backend enhancements focused on data integrity and financial clarity, with the addition of an archived flag for auction items, implementation of minimum bid increments, and clear differentiation between liquid and asset balances for both sellers and buyers. Furthermore, users now have the ability to manage their accounts with profile updates and account deletion, and significant effort was invested in organizing project documentation.

4.4 Sprint 4

Sprint 4 Tasks			
Tasks	**Points**	**Assignees**	**Total Points**
- Bid settlement (Automatically settle bids) - Seller Settles Auction (Manual finalize auction)	2 3	Arkan	5
- Profile Settings UI Improvement - Organized all documents	2 -	Praewa	2
- Syncing Bids for multiple buyers - JavaScript refactoring - Implement pagination for various features	1 1 2	Jason	4
- Update ”place bid” to consider min increment - Add minimum increment in auction item - Adding funds to account - Buyer Dashboard UI Improvement	1 1 3 1	Steve	6

Summary: This sprint saw the successful implementation of automated bid settlement and the introduction of manual auction finalization for sellers. Key feature enhancements included UI improvements for both profile settings and the buyer dashboard, alongside the crucial addition of minimum bid increments to both the ”place bid” process and auction item setup. Under the hood, efforts focused on improving code quality and testing with JavaScript refactoring. Finally, the ability for users to add funds to their accounts was introduced, and the groundwork for scalability was laid with the implementation of bid synchronization for multiple buyers and pagination across various features.

4.5 Points summary

User stories points completion				
Sprint	**Jason Le**	**Praewa Pitiphat**	**Arkan Abuyazid**	**Steve Wang**
1	2.5	1	1.5	3
2	6.5	4.5	3	3
3	8.75	5	5	4
4	4	2	5	6
Total points	21.75	12.5	14.5	16

5 Customer Meetings

5.1 Sprint 1

1. **Meeting 1**
   On 1/30 at 2 PM on Zoom.
   Summary: The initial meeting with the client focused on introducing the team and discussing key product features. The client's primary feedback centered around real-time updates for bidding history and account balances. The team agreed to prioritize core functionality over polished UI initially. The client also suggested implementing separate "real" and "item" account values and recommended researching compatible PostgreSQL versions on the latest Heroku stack. The immediate goals are to set up the Heroku environment and develop a functional login page using Rails.

2. **Meeting 2**
   On 2/6 at 10 AM on Zoom.
   Summary: The second client meeting was a progress check. The client reviewed the implemented functionality and provided positive feedback.

5.2 Sprint 2

1. **Meeting 1**
   On 2/13 at 10 AM on Zoom.
   Summary: This meeting served as a progress check-in with the client regarding the All Pay Auction project. We provided an update on the current development status, highlighting the successful implementation of user/seller account creation and login functionality.

   The client expressed satisfaction with the progress made to date. They offered some suggestions regarding database implementation. While these suggestions were valuable, the client acknowledged that the final decision regarding implementation rests with the development team. We will evaluate the feasibility and potential impact of these suggestions and determine the best course of action.

2. **Meeting 2**
   On 2/20 at 10 AM on Zoom.
   Summary: The second client meeting was a progress check. The client reviewed the implemented functionality and provided positive feedback.

5.3 Sprint 3

1. **Meeting 1**
   On 2/27 at 10 AM on Zoom.
   Summary: The client meeting highlighted several key action items and provided positive feedback on the project's progress. The client requested the implementation of Google login integration, improvements to homepage navigation, stronger password verification, UI enhancements to the buyer dashboard, consistent currency formatting, proper auction image display, refined auction status visibility, direct image upload for sellers, adjusted bid terminology, and default auction date settings. They also requested additional buyer accounts for multi-account testing. Positive feedback was given on the overall design, auction screen layout, bidding flow, and the project's momentum. The primary goal is to complete full transaction functionality by Thursday, March 6th, before the upcoming break, allowing subsequent focus on the action items and remaining tasks.

2. **Meeting 2**
   On 3/6 at 10 AM on Zoom.
   The second client meeting was a progress check. The client reviewed the implemented functionality and provided positive feedback.

5.4 Sprint 4

1. **Meeting 1**
   On 3/27 at 10 AM on Zoom.
   Summary: The client meeting highlighted several key action items and provided positive feedback on the project's progress. The client requested the implementation of ”Auction settlement” as the main goal for this sprint.

2. **Meeting 2**
   On 4/3 at 10 AM on Zoom.
   Summary: The second client meeting was a progress check. The client reviewed the implemented functionality and provided positive feedback.

6 Design Diagram

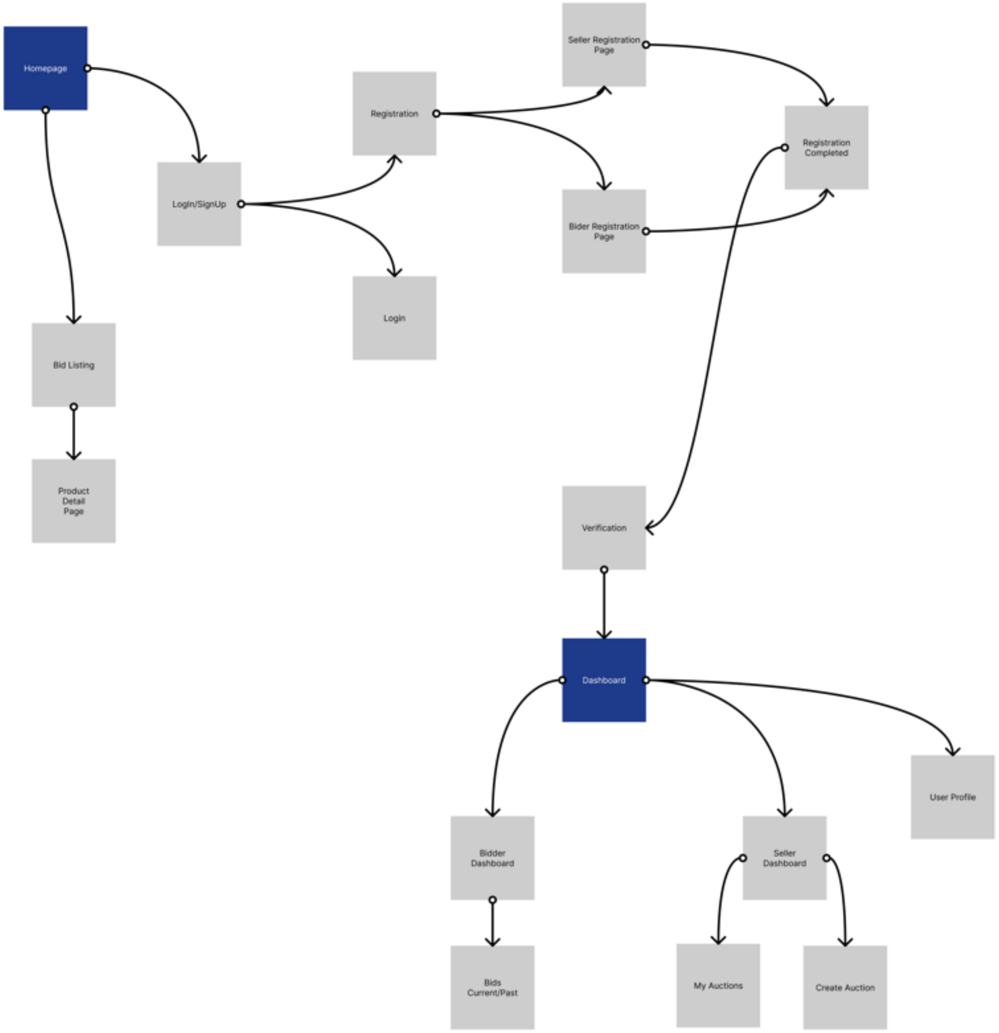

Figure 1: Design Diagram

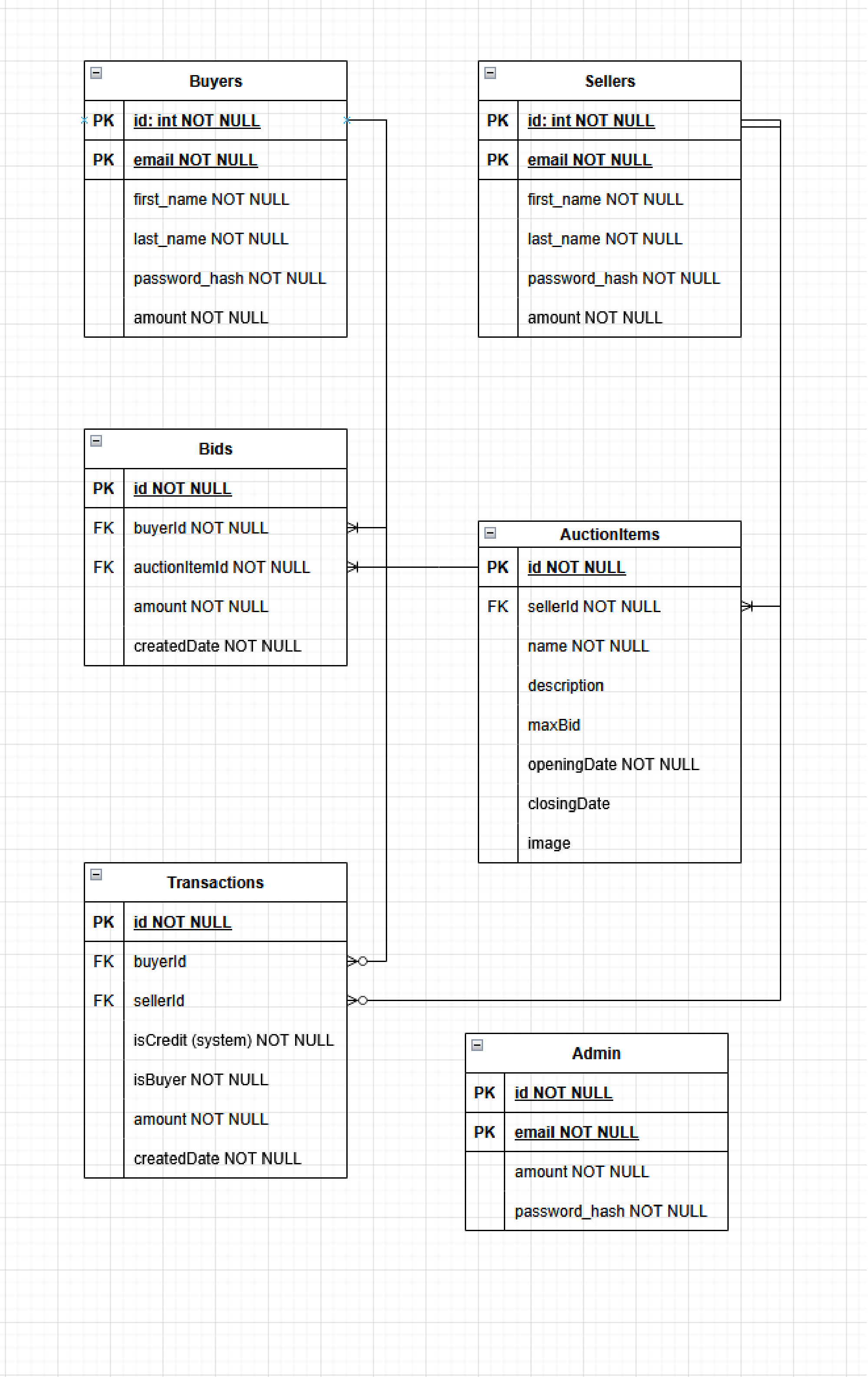

Figure 2: Database Design

7 Tools

- **Version Control**: git, GitHub
  Benefits:
  - Collaboration: Facilitates teamwork by allowing multiple individuals to contribute to the same codebase simultaneously.
  - Change Tracking: Records every modification made to the code, allowing you to revert to previous versions if needed.

  Problems:
  - Learning Curve: Git can have a steep learning curve, especially for beginners, with its numerous commands and concepts.
  - Merge Conflicts: When multiple developers make changes to the same lines of code, merge conflicts can arise, requiring manual resolution
- **Code Coverage**: SimpleCov, CodeClimate
  Benefits:
  - Identifies Untested Code: Clearly shows which lines and branches of your code are not covered by your tests.
  - Improves Test Suite Completeness: Helps you write more comprehensive tests by highlighting gaps in your testing strategy.

  Problems:
  - Can Be Time-Consuming to Achieve High Coverage: Writing tests to cover every line of code can be time-intensive.
  - Focus on Quantity over Quality: High code coverage doesn't necessarily mean your tests are good or cover all important edge cases. You can have high coverage with superficial tests.
- **Ruby gems**: In this project, we use multiple ruby gems such as: `rubocop, stripe, sqlite3`
  Benefits:
  - Extensive Ecosystem: The Ruby community has created a vast and diverse collection of gems for almost every imaginable task, from web development (Rails, Sinatra) and testing (RSpec, Minitest) to background processing (Sidekiq, Delayed Job) and data analysis. This rich ecosystem significantly accelerates development by providing ready-made solutions.
  - Code Reusability: Gems promote code reuse, preventing developers from having to write common functionalities from scratch. This saves time, reduces development effort, and minimizes the potential for introducing bugs.

  Problems:
  - Dependency Hell: Projects can become heavily reliant on a large number of gems, creating complex dependency graphs. Conflicts can arise when different gems require incompatible versions of their own dependencies. Bundler helps mitigate this, but resolving complex dependency conflicts can still be challenging.
  - Maintenance Burden: While the community often maintains gems, some gems might become abandoned or poorly maintained. Relying on such gems can become problematic if bugs are discovered or if they become incompatible with newer versions of Ruby or other dependencies.

8 Repository Organization

The GitHub repository contains the following key components:

- **Application Code**: This includes all the Ruby on Rails code for the `All_Pay_Auction` application. Key directories include:
  - `app/`: Contains the core application logic, including models, views, and controllers.
  - `config/`: Contains configuration files, such as database settings, routes, and environment configurations.
  - `db/`: Contains database migration files.
  - `lib/`: Contains any custom libraries or modules.
  - `Gemfile` and `Gemfile.lock`: Specify the Ruby Gems (libraries) required for the application, ensuring consistent dependencies across different environments.
- **Database Configuration**: The `config/database.yml` file contains the configuration for connecting to the PostgreSQL database.
- **Deployment Configuration:**
  - `Procfile`: Specifies the commands required to run the application on a platform like Heroku.
- **Testing Suite**:
  - `spec/`: Contains RSpec test files for unit and integration testing.
  - `features/`: Contains Cucumber feature files for behavior-driven development (BDD) testing.
- **Docker Configuration**:
  - `Dockerfile`: Defines the environment for Docker, ensuring application can be run in an isolated container.
  - `docker-compose.yml`: Defines multi-container Docker applications.
- **Documentation**:
  - All documentations related to the project are stored in `documentation/`.

8.1 Docker Deployment Process (For testing and development)

run `docker compose build --no-cache` and `docker compose up` to start the app.

9 Discussion

1. Explain your BDD/TDD process, and any benefits/problems from it.
   We use Cucumber for BDD and RSpec for TDD. We first defined a high-level feature behavior (e.g., ”As a user, I should be able to login), then RSpec is used to define the lower-level behavior of individual components (e.g., ”The `User` model should authenticate with a valid password”). This combination helps ensure that the software meets both the user's needs and the required code quality.

2. Discuss your configuration management approach. Did you need to do any spikes? How many branches and releases did you have?
   No spike. We have had several branches since there are many features to be implemented. No official release, but there are 5 Tags in total (for each sprint).

3. Discuss any issues you had in the production release process to Heroku.
   Our team had a few issue with Heroku scheduler (we use this to settle an auction), but the issue has been resolved.

10 Links

1. Github Project: https://github.com/orgs/SIRL-TAMU/projects/1
2. Github Repository: https://github.com/SIRL-TAMU/all-pay-auction
3. Heroku Deployment: https://all-pay-auction3-478f271f12bd.herokuapp.com/
4. Presentation and Demo: https://youtu.be/BzQw5r_CLag?si=d8lIEsjW0X_ZgvcF