



Final Report

Prepared for: CSCE 606

Software Engineering

Prepared by: Team Dot Slash Execute December
9, 2015

Project Name: Auctioneer Contest Scoring System

Team Members: Abhishek Nayyar, Anavil Tripathi, Anamika Bedi, Haitao Ji, Mallika Pokharel, Kiran Yalasangi

TABLE OF CONTENTS

ABSTRACT 2

INTRODUCTION 2

IMPLEMENTATION..... 3

 Judge Perspective: 4

 Administrator Perspective: 4

STAKEHOLDERS 5

USER STORIES 6

 Iteration 0 6

 Iteration 1 7

 Iteration 2 7

 Iteration 3 8

 Iteration 4 8

LO-FI MOCKUPS AND ORIGINAL SCREENSHOTS 9

PROJECT TIMELINES AND MEETINGS..... 13

DATABASE DESIGN AND TABLES 14

TDD/BDD 15

CONFIGURATION MANAGEMENT 17

TOOLS / GEMS USED 17

IMPORTANT LINKS 18

ACKNOWLEDGEMENT 18

REFERENCES 18

ABSTRACT

The Project aim was to facilitate the contests held by Texas Auctioneers Association so as to ease the process and reduce the manual effort. For this we developed an application to solve the afore mentioned problems based on the principles of Software as a Service using the Agile methodology of Software development. The major objective of the software was to provide an online platform for judges to score and comment on the participant performances for each category and for the administrator to manage the enrollment and qualification of participants into respective competitions and rounds. For efficient project development we used tools such as Ruby on Rails (framework), Github (for easier collaboration and version control), Heroku (for easy deployment), RSpec and FactoryGirl (for testing) and other such tools. The complete application was successfully developed and updated in versions, which was finally demonstrated to the project customer to his complete satisfaction.

INTRODUCTION

AUCTIONEER CONTEST SCORING SYSTEM

Current practice for scoring the participants involves extensive use of paperwork by the judges and administrators. This also requires them to employ multiple scorers to calculate, verify and crosscheck the results. The current general structure of the contest has four competitions where each competition has 2-3 rounds. There are around 50 participants and 6 judges per competition. So, on an average the number of paper sheets generated only by scoring would be 3000. Then for score calculation additional enormous amount of paperwork is needed which also introduces a possibility for human errors at numerous places. This even made it necessary for the competition to be held for two day duration whereas only one day is sufficient for conducting the actual contest.

Therefore, the need of the hour was that there should be some automated mechanism for contest management and score calculation. Thus our project aimed at making this cumbersome process into a smooth, error free one. Our approach was to solve the problem in the most generic way possible to increase the flexibility, and to make it feasible for the future changes to be incorporated easily. So, we have provided the features of addition edit and delete for all the entities of our application structure.

Contrast with Iteration 0

We had outlined the following goals during the iteration 0:

- ☑ We will divide our entire SaaS in two parts, one for handling judges record which will be a mobile friendly website making REST calls to our Rails background and another desktop friendly version for administrator manager to manage participants and judge data for different rounds.

- ☑ We would be using common model for managing database and maintaining uniformity between judge and admin data.
- ☑ Our implementation would be making schema for segregating database for participants ID, results and comments for judges and complete information of participants for administrator management. This would ensure the implementation of requirement 3 by client.

We achieved all the goals set by us, over the semester. Due to our flexible model we were able to incorporate some of the modifications that were later added by the client like:

- ✓ Giving contestant number for each round during qualification
- ✓ Question specific comments to each participant while scoring
- ✓ Restricting score modification by judges

IMPLEMENTATION

Our SaaS application is based on MVC architecture with various controllers implemented for rendering data on corresponding view and fetching the data from model. As our application has multi-user access application, so main idea for design architecture is using sessions, differentiated by specific unique session IDs. By using this approach we have ensured that we can reuse same view and controllers for rendering various outputs based on user, this implementing the DRY functionality of software design.

Our Application design can be segregated as two separate design-flows with corresponding features:

- Judge Perspective Design-flow
- Administrator Perspective Design-flow

This application design is formulated on basis of following customer requirements:

- ☑ Create a mobile application/website interface for judges to give score to participants after login authentication.
- ☑ Identity of participants should not be disclosed and judges should be able to score participants on the basis of participant ID.
- ☑ Judge should be able to rate each participants on multiple parameters.
- ☑ Create a website interface for admin to manage entire actions like adding new participants, selecting winners for each round and generating final report for each participants which includes judges scores, comments and final results.
- ☑ Admin should get permissions to manage (modify/add) record in case of conflicts.

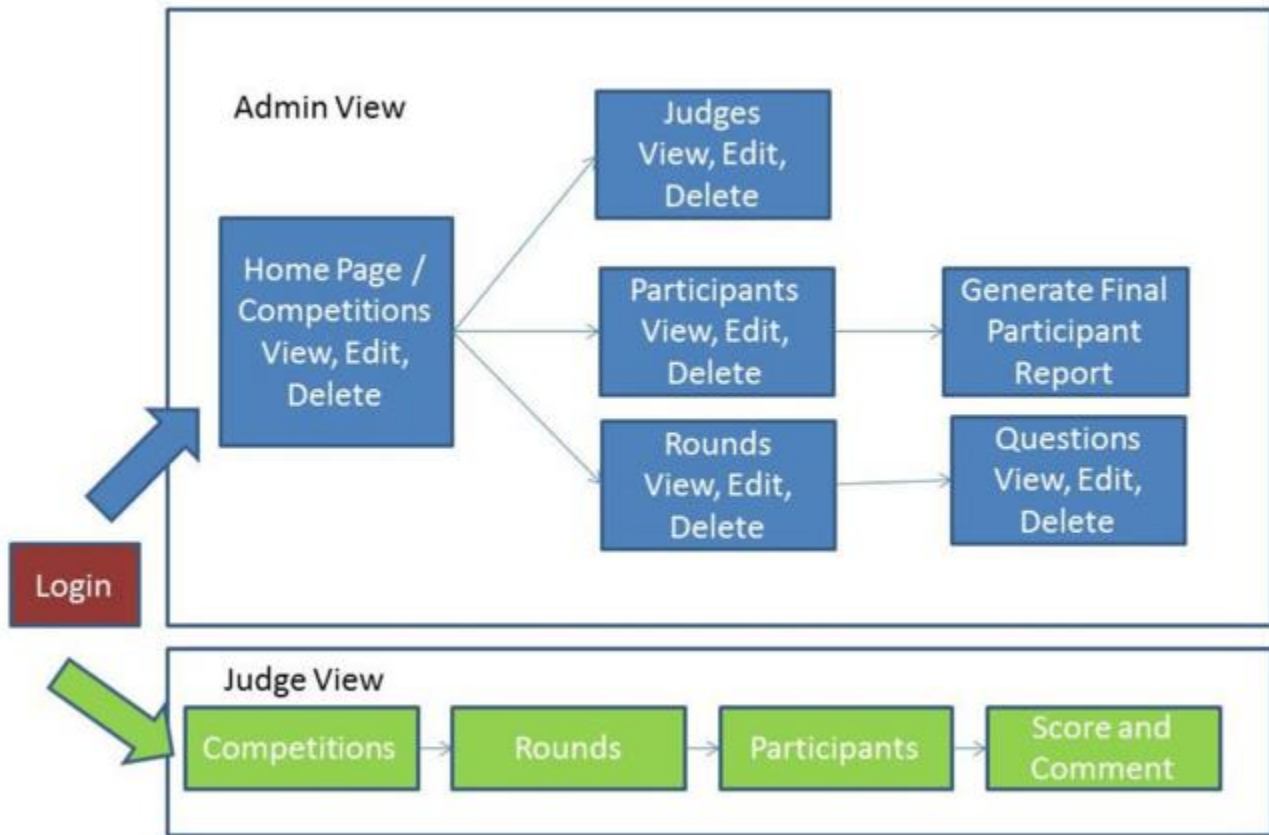


Figure 1. Architecture of our SaaS application

Architecture design shown in Figure 1 highlights two important perspectives:

Judge Perspective:

Judge can access application through security credentials and would be only entitled to view information allowed in his session domain, i.e., he can't check any participant related data, and can't modify any information other than giving scores and comments for a unique participant number. He can't modify score once has submitted final scoring. The modification functionality is only available to super administrator for the case of resolving conflicts.

Administrator Perspective:

Administrator has full access to this application after security authentication. He can modify or add any data, or even check score for any participant. For ease of operation our SaaS has provided administrator with some graphical analysis and content segregation mechanism. These added functionality would ease the workload of administrator. We have provided flexible application design, such that in future if requirement changes, user can register itself and do payment remotely, rather than have to depend on admin for this case. The most critical feature which would ease the work of admin is automated report generation mechanism which will generate report for any participant for all registered data.

STAKEHOLDERS

Customer:

Company: Texas Auctioneers Association

Name: Lance Swigert

Designation: First Vice President

Email : lance@swicoauctions.com

Developers and Team Roles:

- Abhishek Nayyar (Scrum Master)
- Anavil Tripathi (Product Owner)
- Anamika Bedi
- Kiran Yalasangi
- Mallika Pokharel
- Haitao Ji

End User:

Administrator and Judges of Texas Auctioneers Association

Job Rotation:

During the development of entire project, we strongly believed in giving each member equal job responsibility, so we had job rotation for role of scrum master but we kept product owner fixed during entire project in order to ensure level of understanding between product owner and customer which was developed from the start of project.

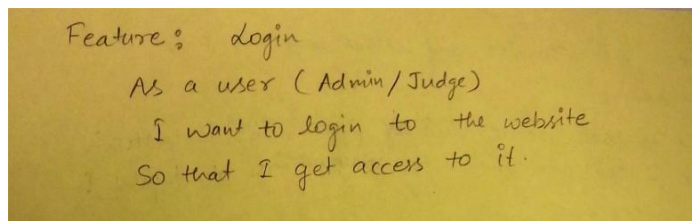
Iteration	Scrum Master	Product Owner
Iteration 0	Abhishek Nayyar	Anavil Tripathi
Iteration 1	Abhishek Nayyar	Anavil Tripathi
Iteration 2	Kiran Yalasangi	Anavil Tripathi
Iteration 3	Anamika Bedi	Anavil Tripathi
Iteration 4	Mallika Pokharel	Anavil Tripathi

The Scrum masters and Product owners for different Iterations have been mentioned in the table above. Apart from following agile methodology of mechanism for having a fixed scrum master for an iteration, we had created a special role for monitoring the overall progress of the project to ensure smooth work transitions. This crucial responsibility was taken up by Haitao Ji. Anavil Tripathi acted as the product owner was our product owner throughout the project for all any customer interactions and for keeping track of customer requirements. We preferred keeping this role fixed for client's convenience.

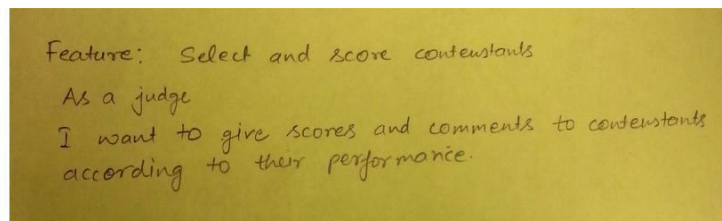
USER STORIES

Iteration 0

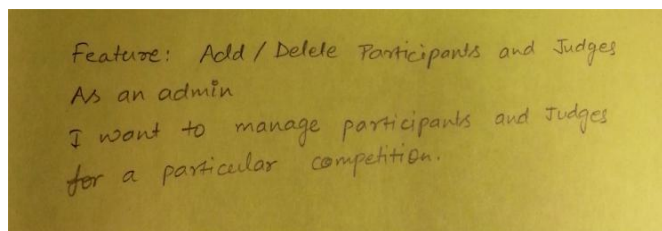
In this iteration, we collected the user stories needed for the basic setup of the contest holding application. We designed the database structure in this period and created the basic architecture of the application. We also finalized the lo-fi UI mockups for both the judge and admin perspectives. We also created an initial application structure that was pushed on Github. User stories were added to pivotal and assigned to each team member. Following are the initial user stories:



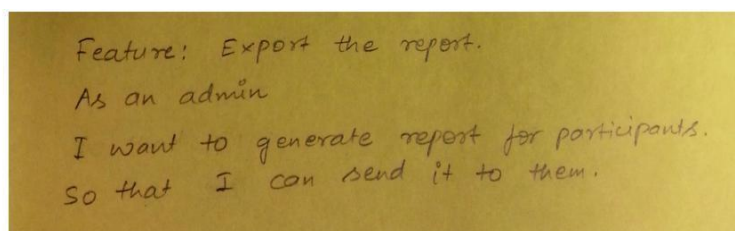
Login



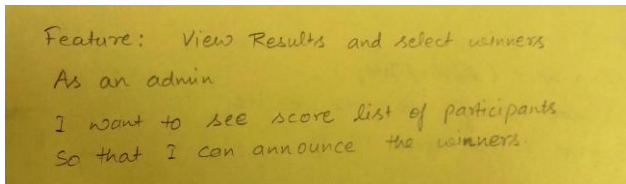
Score contestants



Add/Delete participants



Report generation



Select participants for next round

Iteration 1

We had to add some user stories that were not planned beforehand such as adding, editing competitions and rounds. Following user stories were implemented in this iteration:

- ☑ **Add/ Edit Competitions (3 points):** As an admin I want to add and manage different competitions in the contest
- ☑ **Add/ Edit Participants (3 points):** As an admin I want to add and edit participants in different competitions. So that I can view their details on every competitions.
- ☑ **Add/Edit rounds as an admin (3 points):** I want to add rounds to each competitions so that I can modify rounds with different participants.

Iteration 2

For this iteration we had basic contest structure ready from the previous iteration. So, we had to add some user stories that were not planned beforehand such as adding, editing competitions and rounds. Following user stories were implemented in this iteration:

- ☑ **Login (5 points):** I want to authenticate and login to the website so that I can get access to the website's pages.
- ☑ **Add/Edit Questions (3 points):** As an admin I want to add and edit questions in different rounds so that judge can score the participants for that question.
- ☑ **Add/Edit Judges (5 points):** As an admin I want to add new judges and assign them to correct competitions so that judge can evaluate participants for those competitions.
- ☑ **Add Enrollments and Qualifications (8 points):** As an admin I want to add participant enrollments to desired competitions, and I should be able to add qualified participants to next round. So that judge can evaluate participants for those competitions and rounds.

Iteration 3

In this iteration, we were only concerned with adding the features for scoring part for the judges and to view the results from the admin perspective. For this following user stories were implemented:

- ☑ **Select and score contestants (5 points):** As a judge I want to select contestants and score them so that the scores can be viewed by admin for qualification.
- ☑ **View Results and select qualified participants (3 points):** As an admin I want to view result of a round and select participants for the next round so that qualified participants can participate in next round.

Iteration 4

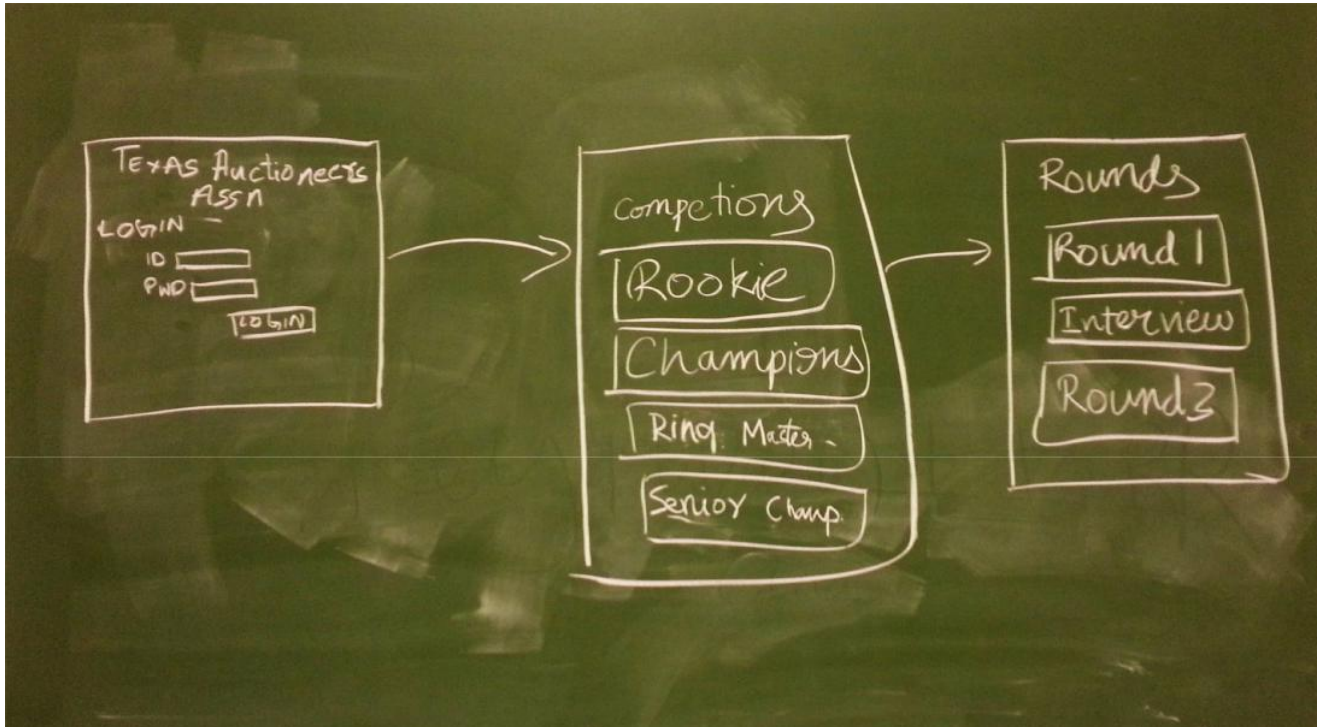
In this iteration, we had to finalize the application for final handover to the client. Following user stories were finished during this iteration.

- ☑ **Addition of comments by judges (3 points):** As a judge I want to give comments to participants on the basis of their performance so that the participants can view the comments later in the report.
- ☑ **Adding validations (3 points):** As a user (admin/ judge) I want to make sure that my inputs are validated so that invalid entries are not created.

By the end of iteration 4, we had an application that completely automated the contest process with complete seed data for client handover.

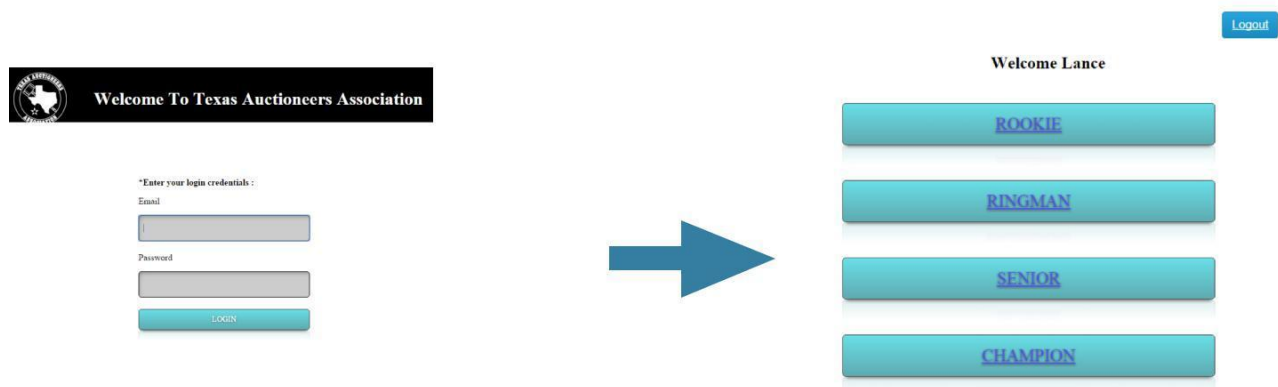
LO-FI MOCKUPS AND ORIGINAL SCREENSHOTS

MOCKUP



This is judge view of project, where after login judge can see different competitions and can then select sound for scoring.

SCREENSHOT:





Rounds

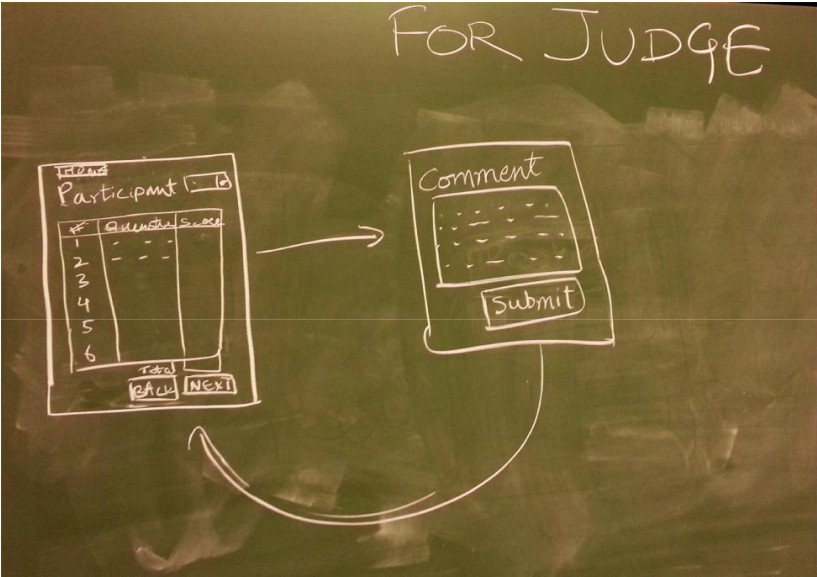
You have gone full screen.

PRELIMS

SEMIFINALS

FINALS

MOCKUP



SCREENSHOT:

Score Participant

Logout

Category	Description	Score	Comment
Opening Statement	Auctioneer Greeting, States Their Name, Contestant Number, and describes the Item for Sale.	<input type="text"/>	<input type="text"/>
Style	Delivery, Poise, Eye Contact, and Gestures.	<input type="text"/>	<input type="text"/>
Overall Bid Calling	Voice control, Voice Clarity, Volume, Speed, Rhythm, Bid Escalation	<input type="text"/>	<input type="text"/>
Professional Image	Appearance, Manner and Attitude.	<input type="text"/>	<input type="text"/>
Salesmanship	Encourages the audience to bid, did bid reach a minimum \$50.00 (no extra credit for above \$50); reached final bid in appropriate time; says "Sold"; announces the bidder number; and repeats the final sale price.	<input type="text"/>	<input type="text"/>
Professionalism	Overall impression: Do you believe the Auctioneer fairly represented the merchandise, the auction profession, and would you hire this Auctioneer to sell your sale.	<input type="text"/>	<input type="text"/>

SUBMIT SCORES

You are at Champion's Prelims round



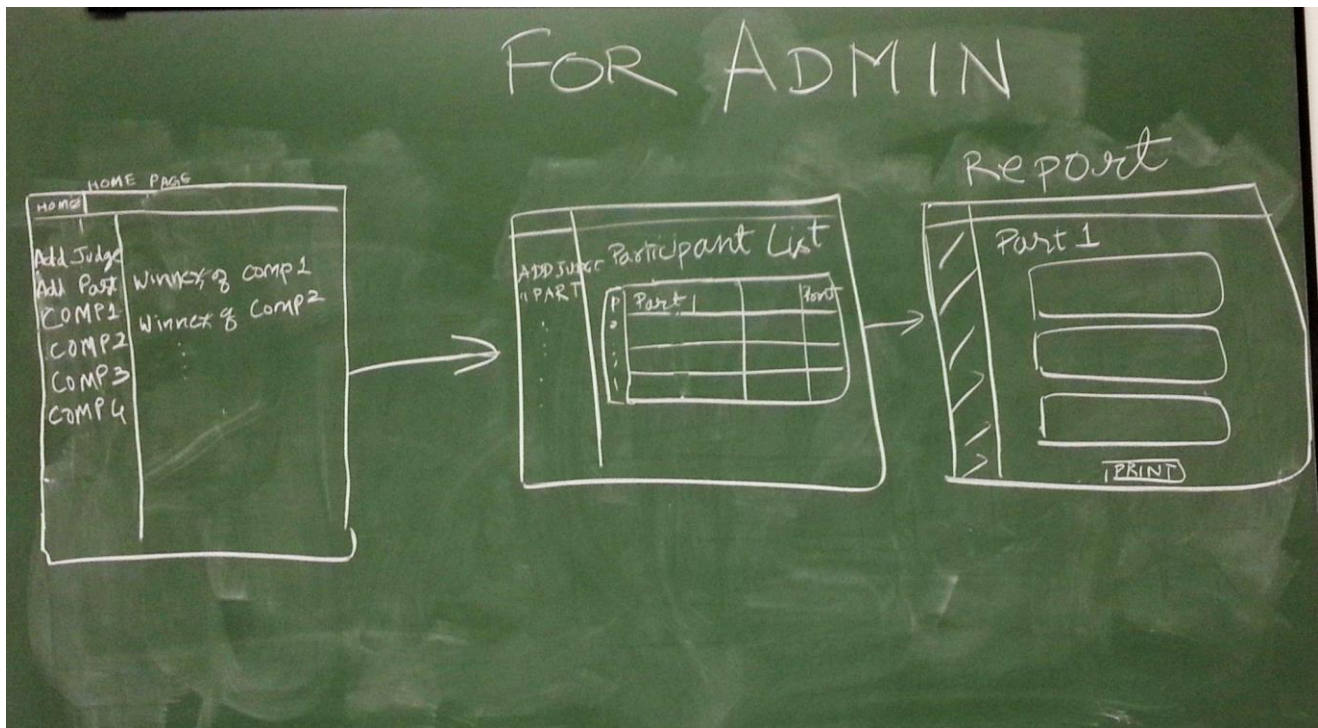
Add Comments

Comment description

There is a lot of scope for improvement

COMMENT

MOCKUP



SCREENSHOT

Competition page :






[Logout](#)

No of competition to display ▼

Type	Name	Description	Edit	Delete
	Rookie	For rookie		
	Ringman	For ringman		
	Senior	For senior		
	Champion	For champion		


Add New Competition

See All Participants



Judges

Rounds for Champion:














Add Round



[Logout](#)

List of rounds for : Champion

Type	Name	Description	Participants	Edit	Delete
	Prelims	First round	6		
	Semifinals	Second round	0		
	Finals	Third Round	0		

Report page for participant :



Final Report

Participant: Anavil

Competition: rookie

Round: Round 1 Judge: Judge Abhi Total Round Score: 60
 Comments for this round: Overall comment

Category	Question	Obtained Score	Comment
Opening Statement	Auctioneer Greeting, States Their Name, Contestant Number, and describes the Item for Sale	10	
Style	Deliver, Poise, Eye Contact, and Gestures	10	
Overall Bid Calling	Voice Control, Voice Clarity, Volume, Speed, Rhythm, Bid Escalation	10	
Professional Image	Appearance, Manner, and Attitude	10	
Salesmanship	Encourages the audience to bid, did bid reach a minimum \$50.00 (no extra credit for above \$50); reached final bid in appropriate time; says "Sold"; announces the bidder number; and repeats the final sale price	10	
Professionalism	Overall impression: Do you believe the Auctioneer fairly represented the merchandise, the auction profession, and would you hire this Auctioneer to sell your sale	10	

Round: Round 1 Judge: Lance Total Round Score: 0
 Comments for this round: *NO COMMENTS*

Category	Question	Obtained Score	Comment
----------	----------	----------------	---------

PROJECT TIMELINES AND MEETINGS

List of customer meeting dates, and their description:

Date: 25th September 2015

Description: On this meeting Iteration 0 user stories were discussed, and an overall functionality of the Project was discussed and how it will help to resolve the problem of assignment.

Date: 16th October 2015

Description: In this meeting, Iteration 1 user stories were discussed (associated with competition, participants, and rounds). A brief demo of the initial flow structure was shown to Mr. Lance Swigert.

Date: 27th October 2015

Description: In this meeting, Iteration 2 user stories were discussed (associated with judge, enrollments and qualifications). A demo was shown to Mr. Lance Swigert of the current iteration where admins could enroll and qualify participants.

Date: 10th November 2015

Description: In this meeting, Iteration 3 user stories were discussed (scoring and viewing results). A brief demo of this iteration was shown. Major portions of the user stories were completed and happy paths were all shown to the client.

Date: 1st December 2015

Description: In this meeting, final functionality of the report generation and possible failure handling and validations were discussed. This meeting finalized the features for this project, and discussion on over future enhancements were also discussed.

DATABASE DESIGN AND TABLES

We have segregated data among different tables and for each table there are relationship with other tables through an intermediate table known as relationship table. Each table has primary key, for identifying it's own entries and foreign key for identifying relationship table entries associated with it.

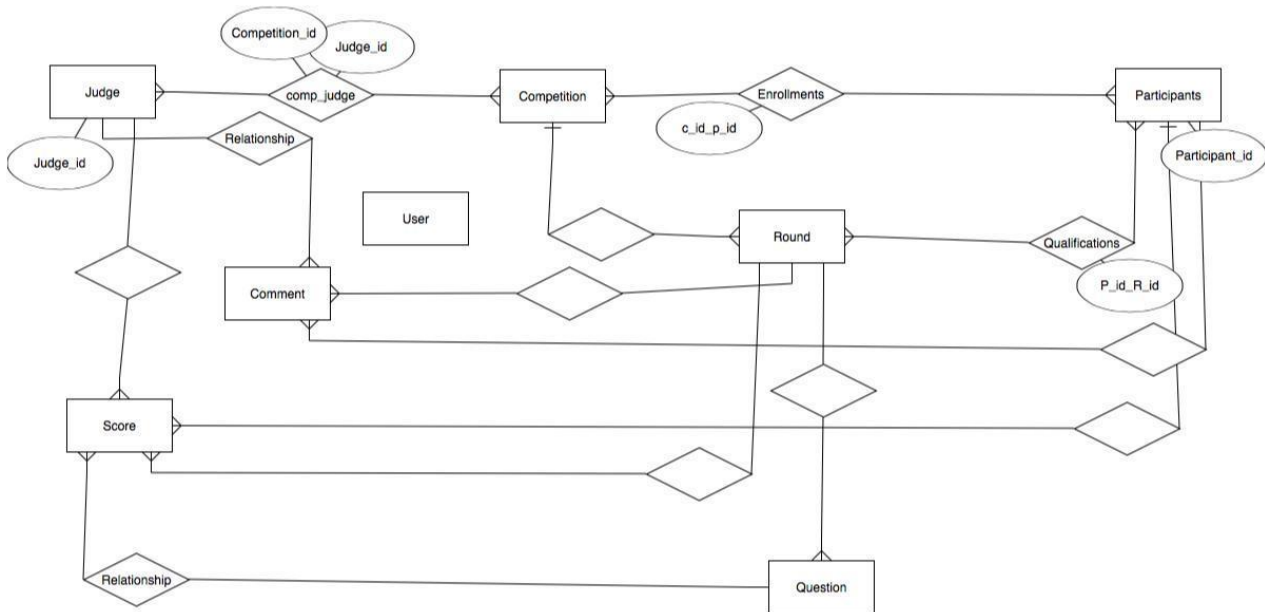
Each controller is associated with particular model for handling data, so before understanding table design for our project, we want to highlight the list of controllers for our project:

Table Listing:

Judge Judge_id Name Email Password	Participant Participant_id Name Email Address	Round Round_id Name Competition_id (U)
	Competition Competition_id Name	Question Question_id Round_id (U) Category Description
Score Score_id Question_id (U) Round_id (U) Judge_id (U) Participant_id (U) Marks Score_Comments	User User_id Email Password Is_admin	Overall Comment Comment_id Comment Competition_id (U) Round_id (U) Participant_id (U) Judge_id (U)

List of controllers
 Competition
 Round
 Question
 Participant
 Comment
 Judge
 Score
 Enrollment
 Qualifications
 competitions_judges
 user
 welcome

Associated Relationship:



TDD/BDD

Test Driven Development or TDD is an advanced technique of using automated unit tests to drive the design of software and force decoupling of dependencies ^[1]. This technique is highly used in Agile development methodologies. The motto of TDD is “Red, Green and Refactor”. We used “RSpec” for TDD.

☑ *Red*: Create a test and make it fail

☑ *Green*: Make the test pass by any means necessary

Refactor: Change the code to remove duplication in your project and to improve the design while ensuring that all tests still pass

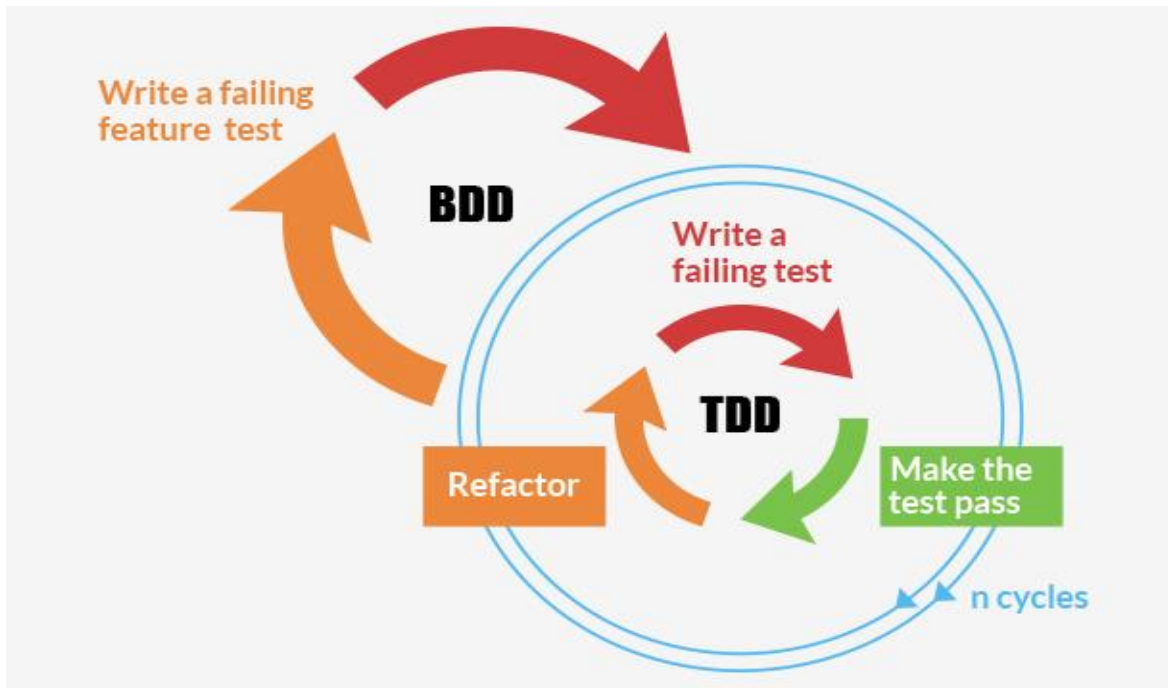


Figure 2. Test development approach and behavior driven approach [Ref: Wiki]

During our project development we relied heavily on these tools for analyzing our test coverage and results. We tried to write the BDD and test cases before writing the actual code in the respective controller and view files so that the desired behavior could be tested as soon as implemented. In this way, both verification and validation were done as soon as the code was written. We merged the code into master for deployment only after it was tested. .

This approach had following advantage during our project development.

- ☑ Enabled us to properly design feature: We were able to locate potential sad cases before development of module, which eased its development. For example during scoring by judges, after scoring was done for some participants, those participants should not be listed anymore.
- ☑ Enabled testing considering worse case scenarios: During development of TDD for the project, we were able to analyze worst case scenarios such as unwanted actions and commands, which could potentially cause some problem(s) in desired output. For example, update call to any judge enrollment for competition should also update properly the competitions_judge table. These checks ensured we had uniform output throughout as desired.
- ☑ Identifying Dead-Code: Proper TDD/BDD helped us to identify and clean up any dead code in our project.

CONFIGURATION MANAGEMENT

For version control we chose git over subversion due to its robustness and better collaboration features, available through Github. We used one main branch “master” for the deployment purposes. For major changes to the project we also created different branches that were later merged into the master branch once they were finalized. Following are some example branches that were later deleted:

- ☑ **Login:** This branch in git was for implementing the login feature during the iteration 2. This was later merged into the master once login was properly functioning.
- ☑ **Qualification:** It implemented the qualification feature for the participants.
- ☑ **TDD:** TDD was added into this branch

Master: This is the main branch of our code. All the above branches are merged into this main branch by resolving all the conflicts. We also raised issues for tracking bugs, enhancements etc for easier handling between team members. By the end of the project all the issues were fixed and closed.

We have four releases of our software, one release per iteration with proper tags assigned to each release. We faced some issues on first deployment to Heroku as we were initially using only sqlite, so later we switched to postgres for Heroku. Most of us had Unix environments, but one member also used virtual box and another used Cloud9 for the development and we didn’t face any major issues.

TOOLS / GEMS USED

Development

- ☑ Ruby 2.2.1
- ☑ Rails 4.2.0
- ☑ Heroku
- ☑ GitHub
- ☑ Rspec3.2.3
- ☑ Cucumber 1.3.19
- ☑ Bcrypt 3.1.2
- ☑ FactoryGirl
- ☑ Simplecov
- ☑ Launchy
- ☑ uglifier
- ☑ For development environment we used: sqlite3, faker, jquery-rails
- ☑ For production these specific gems were used: pg, jquery-rails

Tracking

☒ Pivotal Tracker

Communication

☒ Slack

IMPORTANT LINKS

- Video Link (Introductory Video): <https://vimeo.com/148410575>
- Video Link (Demo Video): <https://vimeo.com/148419294>
- Video Link (Final Video): <https://vimeo.com/148430523>
- Git Repository: https://github.com/abhinavvar/SE_DSE_ACSS.git
- Pivotal Tracker: <https://www.pivotaltracker.com/n/projects/1436478>

ACKNOWLEDGEMENT

We would like to extend our sincere thanks to Dr. Jeff Huang and TA's for helping us throughout the project. Special thanks to our client Mr. Lance Swigert for being supportive. We also express gratitude towards the Department of Computer Science and Engineering, Texas A&M University for all the resources made available to us and for this opportunity.

REFERENCES

1. <http://www.ryangreenhall.com/articles/bdd-by-example.html>
2. <http://code.tutsplus.com/tutorials/the-newbies-guide-to-test-driven-development--net-13835>
3. <https://www.wikipedia.org>
4. <http://www.w3schools.com/css/>