betcha!

Team 19 - Sprint #1 Planning Document:

Kushagra Kushagra, Kyle Ohanian,
Noah Smith, Peter Jones, Siddharth Shah

Sprint Overview

The goal of this first sprint is to create all the system critical databases as well as establishing basic client-server connection methods to allow communication. This will look like the development of the *Users*, *Friend Pairs*, and *Bets* tables in the server database and to create all of the query methods related to creating and viewing the entries in these tables. The data gained from these queries should then be linked to API endpoints and client-server requests to get the data from the server to the Android Application. Finally, the last component of the sprint is the displaying of these system critical components to the user through the creation of a basic UI/UX.

Scrum Master: Peter Jones

Meeting Schedule: Standups: (Wednesdays 4:30pm - 6:00pm, Saturdays 10:00am - 12:00pm)

Risks and Challenges

The main risk of this sprint is the possible accumulation of technical debt. As with the start of any project, it is key to avoid all possible putting off in the initial stages as this only snowballs into future sprints. The challenge for this sprint is to fully think through the implementation of each component and how it might affect the system down the road.

As for technical challenges, chief among these are the creation of the API endpoints and their relative request methods on the client side. This component will take a great amount of communication between the teams. Similarly, the Venmo API integration on both the client and server sides will require a large amount of coordination.

Current Sprint Detail

User Story #1

As a user, I would like to login to my account

Task #	Description	Hours	Team	Owner
1	Create "User" table in server database	2	BE	Sidd
2	Write database connection methods for server	4	BE	Sidd
3	Write server User authentication query methods	3	BE	Sidd
4	Write API logic for User authentication	4	BE	Sidd
5	Create API endpoint for user login	1	BE	Sidd
6	Write Venmo API connection methods	4	BE	Sidd
7	Write server login confirmation methods	2	BE	Sidd
8	Create User login page	3	FE	Kyle
9	Write Venmo Oauth SDK Integration	6	FE	Kyle
10	Write unit tests for modules	1	BOTH	Sidd

Acceptance Criteria

- Given that the user has a Venmo account, when they click on the Login button on the launch screen, they should be authenticated fast and accurately using the Venmo Oauth.
- Given that the client OAuth button grabs a token from Venmo OAuth service successfully, after the user clicks the login button, the token should be posted to betcha! Server, the betcha! Server should check its credibility from the Venmo OAuth service and return appropriate status of authentication.
- Given that the authentication from the server communication between betcha! Server and Venmo OAuth server is successful, when the user clicks on the login button, they should be navigated to the main screen of the app.

User Story #2

As a user, I would like to create an account with Venmo

Task #	Description	Hours	Team	Owner
1	Write server User creation methods to query database	1	BE	Noah

2	Write server Venmo API account confirmation logic	3	BE	Noah
3	Create API endpoint for user account creation	1	BE	Noah
4	Create User account creation page	4	FE	Kyle
5	Write API request methods for account creation	3	FE	Kyle
10	Write unit tests for modules	1	BOTH	Kyle

Acceptance Criteria

- Given that user doesn't have a Venmo account, when they click on login button, the Venmo flow should allow them to create a new account.
- Given that the user has a valid account on Venmo and not on betcha!, when they click login and the Venmo OAuth service generates a valid token, the betcha! Server should create a new entry in the database with user details.
- Given that the user details persist with Venmo and betcha!, when they sign in, the user's first time account should be populated accordingly with available details from Venmo and the open betcha! feed.

User Story #3

As a user, I would like to delete my account

Task #	Description	Hours	Team	Owner
1	Write server User deletion query methods	3	BE	Sidd
2	Create API endpoint for individual user deletion	2	BE	Sidd
3	Create API request methods for User deletion	2	FE	Kush
4	Create Account Deletion button	3	FE	Kush
5	Write unit tests for modules	1	BOTH	Kush

Acceptance Criteria

- Given that the user is logged in, when they click on delete account from their profile, their account record should be deleted from active user table and moved to passive users for keeping consistency in data of past bets they might have taken part in.
- Given that the user has deleted their account, when any other user who was previously connected with them tries to navigate to their profile should get error messages/blank navigation pages.
- Given that the user has deleted their account, when any other user tries to initiate a bet with them, the app shouldn't let them.

User Story #4

As a user, I would like to create custom bets

Task #	Description	Hours	Team	Owner
1	Create "Bets" table in server database	2	BE	Noah
2	Write "Bets" table connection methods for server	2	BE	Noah
3	Create API endpoint for bet creation	2	BE	Noah
4	Create Bet creation page	4	FE	Kush
5	Create "Bets" table in client database for caching	5	FE	Kush
6	Create API request methods for bet creation	2	FE	Kush
7	Write unit tests for modules	1	BOTH	Noah

Acceptance Criteria

- Once the user has logged in, they should see a button to create a custom bet
- Given the user had pressed the create bet button, they should be presented a form to create the bet
- Once the user has created the bet, the bet should be present in the Bets table on the server

User Story #10

As a user, I would like to view my open bets

Task #	Description	Hours	Team	Owner
1	Create server User Bet view query methods	1	BE	Noah
2	Create API endpoint for User Bet viewing	2	BE	Noah
3	Create User Bet viewing page	3	FE	Kyle
4	Create individual Bet viewing page	2	FE	Kyle
5	Create API request methods for Bet viewing	3	FE	Kyle
6	Write unit tests for modules	1	BOTH	Kyle

Acceptance Criteria

- Given that the user is logged in, when they click on the Bets tab on the navigation, they should be able to see all of their current bets and past bets, along with the option to view the descriptions of each bet they are apart of.

- Given that the user has created a new bet, when they press the create button, they should be allowed to see their bet show up on their list and be able to open it and view the contents of the bet.
- Given that a newly created bet is made public, when a friend decides to join a bet, the user will be notified and should be taken straight to the bet viewer to see who joined and other parts.

User Story #17

As a user, I would like to add friends

Task #	Description	Hours	Team	Owner
1	Create "Freind Pairs" table in server database	2	BE	Noah
2	Write server Friend Pair creation query methods	1	BE	Noah
3	Create API endpoint for Add Friend	2	BE	Noah
4	Create add friend page / button	3	FE	Kush
5	Create API request methods for Add Friend	2	FE	Kush
6	Write unit tests for modules	1	BOTH	Noah

Acceptance Criteria

- Given that the user is logged in, when they click on the Add Friend page they will be able to view current friend requests, and be able to search for new friends.
- Given that the searched user was found in the user database, a friend request will be sent to that user if the friend pair was not found in the database. The friend request will be visible within the add friend page of both users until the user accepts or declines the request.
- Given that the user accepts the friends request, the Friend Pairs database will be updated, and the friend request will be removed from the add friend page of both users.

User Story #18

As a user, I would like to remove friends

Task #	Description	Hours	Team	Owner
1	Write server Friend Pair delete query methods	1	BE	Noah
2	Create API endpoint for Remove Friend	2	BE	Noah

3	Create Remove Friend button	1	FE	Kush
4	Create API request methods for Remove Friend	2	FE	Kush
5	Write unit tests for modules	1	BOTH	Kush

Acceptance Criteria

- Given that the user is logged in, when they click on the Remove Friend button they will be prompted with a list of all the user's friends, retrieved from the Friend Pair database.
- Given that the user selects a friend on the list, they will be prompted with a confirmation of their selection.
- Given that the user confirms the deletion of a friend, the Friend Pair is deleted from the database, and the list of friends displayed will be updated.

User Story #19

As a user, I would like to view a feed of bets

Task #	Description	Hours	Team	Owner
1	Write server Bet Viewing methods to query database	3	BE	Peter
2	Create API endpoint for Bet Viewing	3	BE	Peter
3	Create Bet Feed page	4	FE	Peter
4	Create API request methods for Bet Viewing	2	FE	Peter
5	Create Bet caching methods to local database	3	FE	Peter
6	Write unit tests for modules	1	BOTH	Peter

Acceptance Criteria

- Given that the user is logged in, when they click on the Bet Feed page they will see a list of recently complete, and ongoing bets that they or their friends are a part of. This list will display details about the bet, with the option to view bet.
- Given that the user selects to view a bet, the client with query the Bets database and display the User Bet viewing page populated by the selected bet info.
- Given that the user has ongoing or completed bets to be listed, the user will be able to scroll through bets up to one week old.

User Story #20

As a user, I would like to see past bets

Task #	Description	Hours	Team	Owner
1	Write server individual user query methods	3	BE	Noah
2	Create API endpoint for individual user profile access	2	BE	Noah
3	Create API request methods for individual users	2	FE	Kyle
4	Create User Profile page	3	FE	Kyle
5	Write unit tests for modules	1	BOTH	Kyle

Acceptance Criteria

- Given that the past bet is saved in the database, user should be able to view their past bets through their client profile after a request to view has been made by the client app
- Given that the User Profile page has been successfully created, then the user would be able to view bets currently in progress and past bets through their profile page along with amount of payments made in each bets and to who, which is all saved in the database
- Given that the bets have been made successfully by both users, a request to view made by either client should access the same data from the database to maintain consistency when viewing the same bet made by two different users

User Story #28

As a user, I would like to change my password

Task #	Description	Hours	Team	Owner
1	Write server User password update query methods	2	BE	Peter
2	Create API endpoint for password update	2	BE	Peter
3	Create API request methods for password update	2	FE	Peter
4	Create password change button	2	FE	Peter
5	Create password update page	2	FE	Peter
6	Write unit tests for modules	1	BOTH	Peter

Acceptance Criteria

- Given that the password change button would direct the user to the password update change where the should allow the user to change their old password to a new, valid password

- Given that password has been updated by the user, the new password must replace the old password in the database so future login by the user would be successful with the new password
- Given that the password has been updated in the database, an email or test module would be used to confirm the password has been updated and would show in the user profile

User Story #29

As a user, I would like to change my email

Task #	Description	Hours	Team	Owner
1	Write server User Email update query methods	2	BE	Peter
2	Create API endpoint for email update	2	BE	Peter
3	Create API request methods for email update	2	FE	Peter
4	Create email change button	2	FE	Peter
5	Create email update page	2	FE	Peter
6	Write unit tests for modules	1	BOTH	Peter

Acceptance Criteria

- Given that the User has requested to change email address, the new email address must be valid and not taken by another user already
- Given that the user has changed their email address, it should be changed in the database to update it in their profile and for other users who are friends with them
- Given that the email address has been changed, a confirmation email or test module would be used to check if the email entered is correct and what the user intended to enter

Remaining Backlog

1. As a user, I would like to login to my account
2. As a user, I would like to create an account with Venmo
3. As a user, I would like to delete my account
4. As a user, I would like to create custom bets
5. As a user, I would like to send bet to friend
6. As a user, I would like to receive bets from friends
7. As a user, I would like to accept bets
8. As a user, I would like to confirm the completion of bets
9. As a user, I would like to fulfill bets using Venmo
10. As a user, I would like to view my open bets
11. As a user, I would like to edit my open bets
12. As a user, I would like to cancel my open bets
13. As a user, I would like to confirm the cancellation of a bet
14. As a user, I would like to view friends' open bets
15. As a user, I would like to like friends' bets
16. As a user, I would like to comment on friends' bets
17. As a user, I would like to add friends
18. As a user, I would like to remove friends
19. As a user, I would like to view a feed of bets
20. As a user, I would like to see past bets
21. As a user, I would like to see my betting record
22. As a user, I would like to be reminded of bets near the end of their term
23. As a user, I would like to view friends' past bets
24. As a user, I would like to report a technical issue
25. As a user, I would like to join friends' bets
26. As a user, I would like to create groups
27. As a user, I would like to create group bets
28. As a user, I would like to change my password
29. As a user, I would like to change my email
30. As a user, I would like to have my email confirmed