Functional Requirements

1. Make user profile
2. Verify users with Baylor email
3. Tutorial of app when new account is created
4. Separate Rider/Driver feeds
5. Create new Driver posts
6. Create new Rider posts
7. Filter posts by date and location
8. Sort posts by time
9. On new post creation, show suggestions for users who meet requirements
10. Remove rides that have expired (past date)
11. Riders/Drivers can cancel posts
12. Remove canceled posts from feed/waitlists/etc.
13. Rider and Driver must both confirm
14. Remove full rides that are 100% confirmed
15. “Waitlist” riders if post is not 100% confirmed
16. Share contact info when ride is confirmed
17. Push notifications for waitlists, found rides, etc.
18. Set reminders for Riders/Drivers

Actors

- Users - There are Riders and Drivers, a user can be either depending on their needs.
  - Riders: The users that need a ride from point A to point B.
  - Drivers: The users that provide a ride from point A to point B.
- SystemOperations - The operations of the system that updates or generate reports from the system database
- Administrators - The staff that helps maintain the the performance of the system.

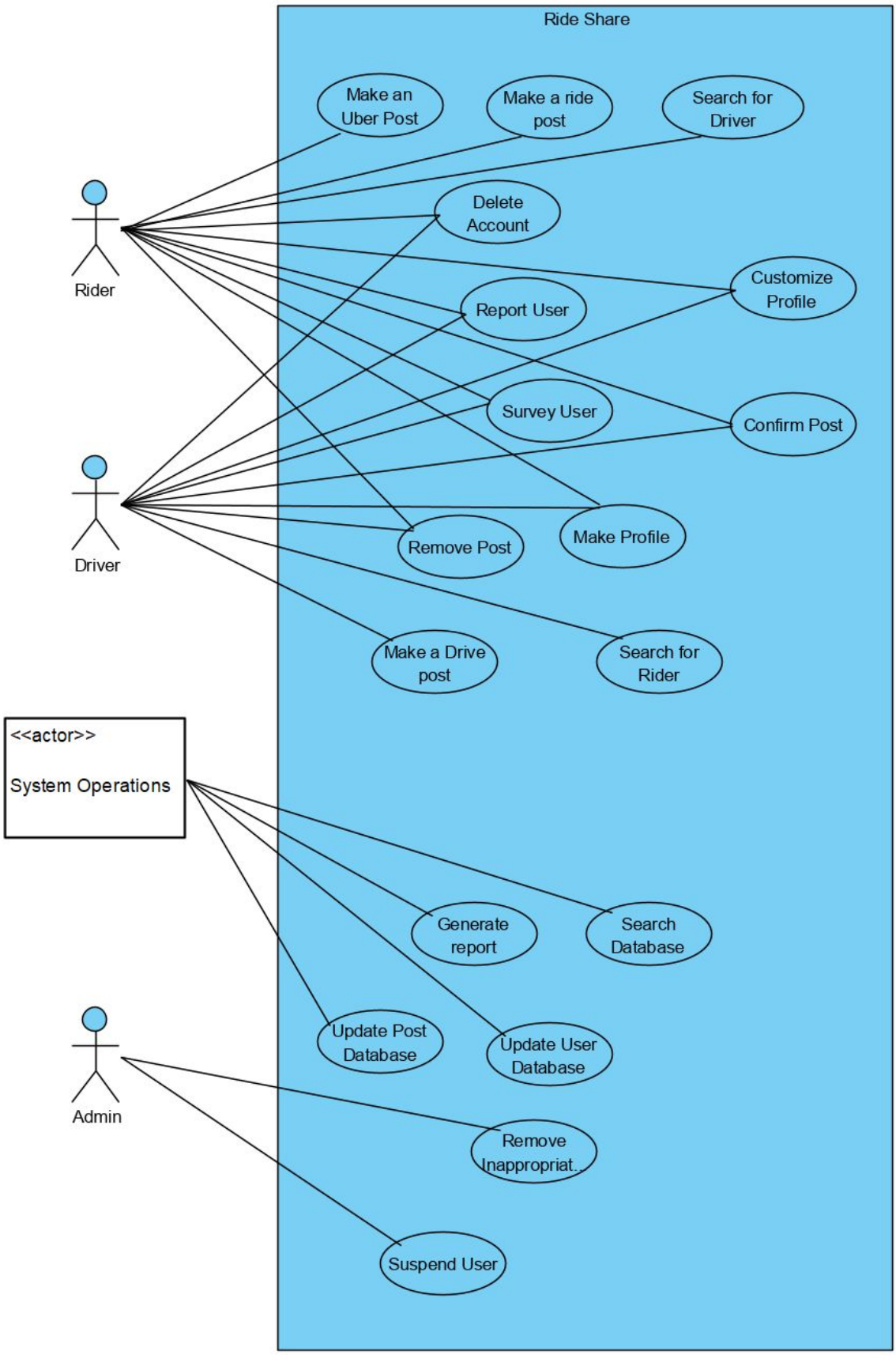
UseCases + System Sequence Diagram:

Black uses cases are for users

Brown use cases are for the admin

Blue use cases are for the system internal operations

1. Make Rider Post - Joseph Y
2. Make Driver Post - Joseph Y
3. Make Uber Post - Joseph Y
4. Report User - Leighton
5. Search for Driver - Josh
6. Search for Riders - Josh
7. Delete Account - Mohsen
8. Remove a Post - Mohsen
9. Making Profile - Leighton
10. Survey User - Joseph P
11. Confirming Post - Joseph P
12. Customize profile - Leighton
13. Suspend User - Mohsen
14. Remove Inappropriate posts - Joseph P
15. Search Database for post - Josh
16. Generate report - Andrew
17. Update Post Database - Andrew
18. Update User Database - Andrew


Ride Share
Make an Uber Post
Make a ride post
Search for Driver
Delete Account
Customize Profile
Report User
Survey User
Confirm Post
Make Profile
Remove Post
Make a Drive post
Search for Rider
Rider
Driver
<<actor>>
System Operations
Generate report
Search Database
Update Post Database
Update User Database
Remove Inappropriat..
Suspend User
Admin

Use Case: Make Rider Post - Joseph Yu

Scope: Baylor Rideshare getting a ride

Level: User Goal

Primary Actor: Rider

Stakeholders and Interests:

- Rider: Wants to find rides available easily.

- Driver: Wants to find riders available easily.

- System Administrators: Wants to satisfy the users.

- Database: Wants to record all ride shares.

Preconditions: The Rider has registered an account on the Baylor Rideshare app.

Success Guarantee (Postconditions): The Post Database is updated.

Main Success Scenario (or Basic Flow):

1. The User opens the application and selects “new ride post.”
2. The system requests information from the rider.
3. The rider enters “point A, point B, date, and time.”
4. The System shows available rides that fit the Rider’s inputs.
5. The Rider selects a ride to join.
6. The System sends a request to the Driver.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application and restart.
    2. The Rider starts a new request to make a new rider post.

3a. The Rider cancels the new ride post:
    1. The System records the cancelled request, the rider request is not posted.

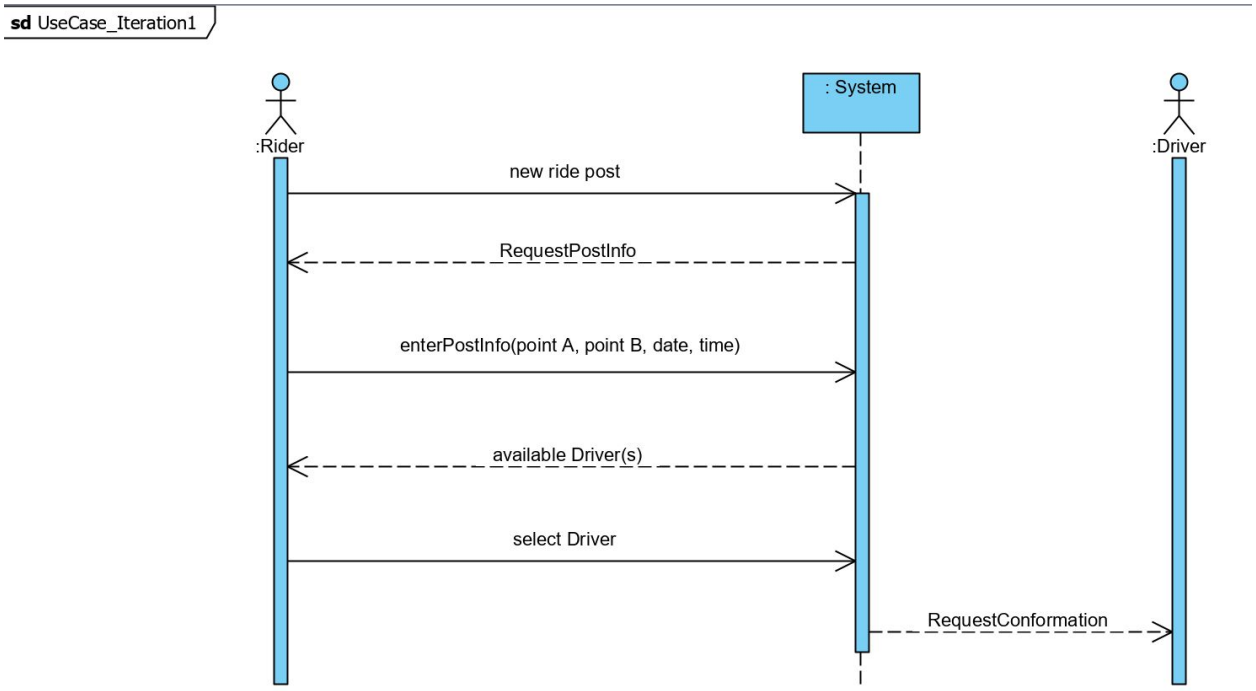
3b. The Rider enters invalid information:
    1. The System indicates that there are errors in the input.
    2. The Rider enters the information again.
    Repeat step 1-2 until the information is valid

4a. There are no available rides that fit the requirements of the Rider:
    1. The System asks if the Rider would want the System to be on the watch for an available ride later on and notify the Rider if one is found.
    2. The Rider agrees and the ride is posted and recorded by the System.
        2a. The Rider cancels the request:

1. The System records the request, the rider post is not posted.

5a. The Rider cancels the new Ride Post:

1. The System records the cancelled request, the rider request is not posted.



Use Case: Make Driver Post - Joseph Yu

Scope: Baylor Rideshare finding riders

Level: User Goal

Primary Actor: Driver

Stakeholders & Interests:

- Rider: Wants to find rides available easily.

- Driver: Wants to find riders available easily.

- System Administrators: Want to record the ride shares and satisfy the needs of the users.

-Database: wants to keep an accurate record of all ride shares.

Preconditions: The Driver has registered an account on the Baylor Rideshare app.

Success Guarantee (Postconditions): The Post Database is updated.

Main Success Scenario (or Basic Flow):

1. The driver wants to provide a ride for potential riders, the driver opens the application and selects “new driver post.”
2. The system requests information from the driver.
3. The driver enters “point A, point B, number of people, date, and time.”
4. The application shows the potential riders that fit the driver’s ride.
5. The driver selects rider(s).
6. The application sends request(s) to the Rider(s).

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application and restart.
    2. The Driver starts a new request to make a new driver post.

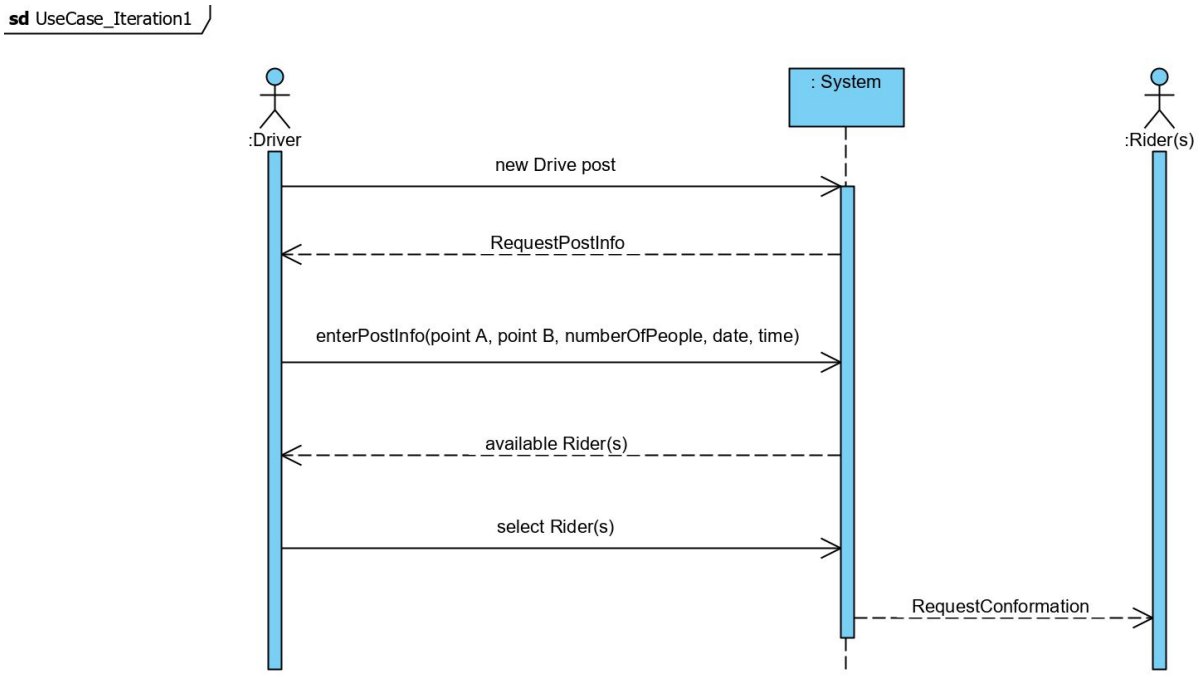
3a. The Driver cancels the new driver post:
    1. The System records the cancelled request, the Driver offer is not posted.

3b. The Driver enters invalid information:
    1. The System indicates that there are errors in the input.
    2. The Driver enters the information again.
    Repeat step 1-2 until the information is valid.

4a. There are no available rider(s) that fit the requirements of the Ride:
    1. The System records the post and posts the new Driver post.

4b. The Driver cancels the driver post:
    1. The System records the driver post offer, and the new driver post is not posted.

5a. The Driver cancels the driver post:
    1. The System records the driver post offer, and the new driver post is not posted.



Use Case: Make Uber Post - Joseph Yu

Scope: Baylor Rideshare finding Uber riders

Level: user goal

Primary Actor: Riders

Stakeholders & Interests:

- Rider: Wants to find rides available easily.

- Driver: Wants to find riders available easily.

- System Administrators: Wants to record the ride shares and satisfy the needs of the users.

Preconditions: The Rider has registered an account on the Baylor Rideshare app.

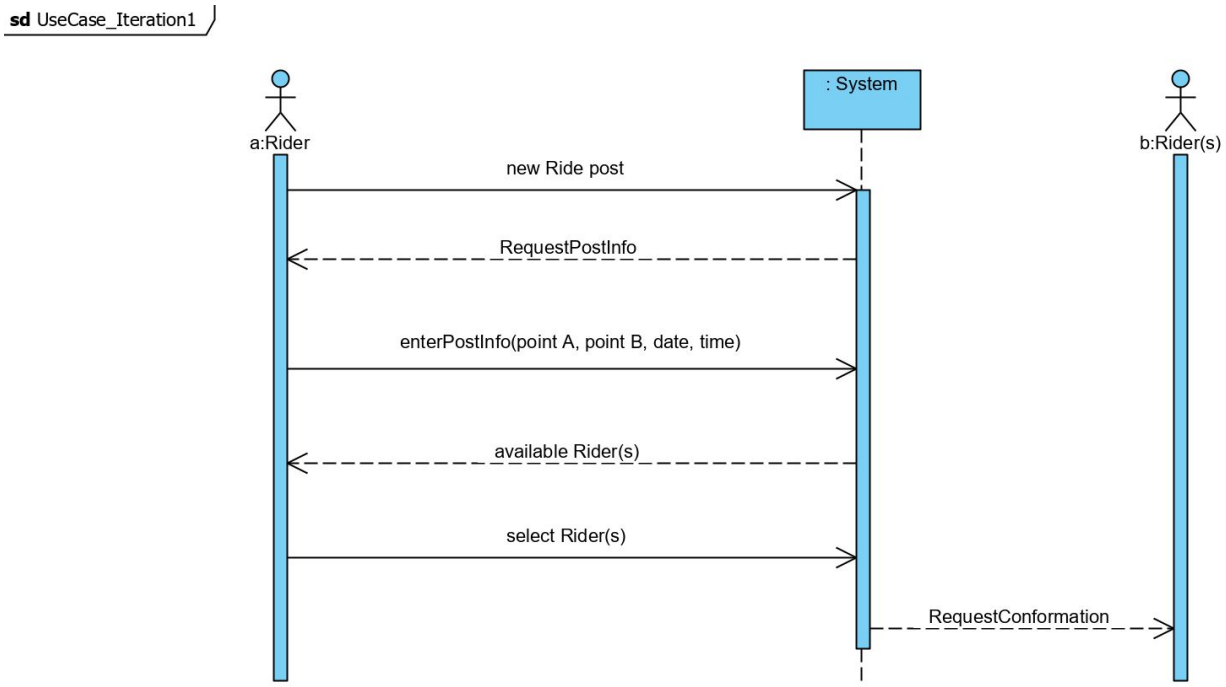
Success Guarantee (Postconditions): The Post Database is updated.

Main Success Scenario (or Basic Flow):

1. The rider requires transportation from point A to point B, the rider opens the application and selects “new Uber ride post.”
2. The system requests information from the rider.
3. The rider enters “point A, point B, date, and time.”
4. The application shows the available rides that fits the Rider’s requirements.
5. The Rider selects a ride.
6. The application sends request(s) to the Rider(s).
7. The Rider closes the application.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application and restart.
    2. The Rider starts a new request to make a new rider post.
3a. The Rider cancels the new ride post:
    1. The System records the cancelled request, the rider request is not posted.
3b. The Rider enters invalid information:
    1. The System indicates that there are errors in the input.
    2. The Rider enters the information again.
    Repeat step 1-2 until the information is valid.
4a. There are no available rides that fit the requirements of the Rider:
    1. The System asks if the Rider would want the System to be on the watch for an available ride later on and notify the Rider if one is found.
    2. The Rider agrees and the ride is posted and recorded by the System.
        2a. The Rider cancels the request:
            1. The System records the request, the rider post is not posted.
5a. The Rider cancels the new Ride Post:
    1. The System records the cancelled request, the rider request is not posted.



Use Case: Report User - Leighton Glim

Scope: Baylor Rideshare reporting drivers/riders

Level: User goal

Primary Actor: Drivers and Riders

Stakeholders & Interests:

- User: Wants to report bad experiences with other users.

- System Administrators: Wants to maintain safety and experiences of users.

Preconditions: The Driver/Rider has registered an account on the Baylor Rideshare app and has partaken in a ride or views unsafe content on the app.

Success Guarantee (Postconditions): The report is filed in the database.

Main Success Scenario (or Basic Flow):

1. User partakes in ride.
2. The user will open the app and select “Report User.”
3. System prompts the user for details.
4. The name of the user they would like to report and reason for reporting are entered.
5. The System then shows a summary of the report.
6. User confirms the report.
7. The System closes the dialog.

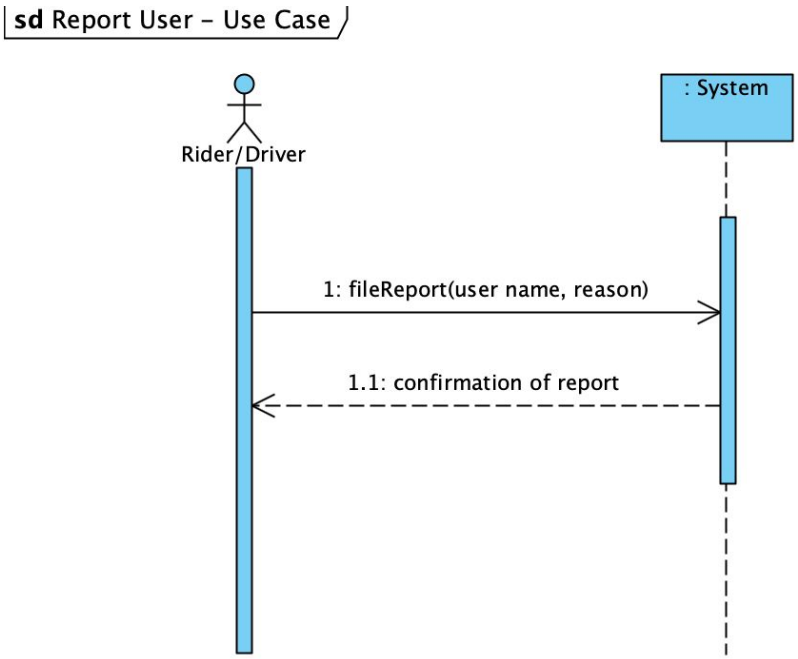
Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application and restart.
    2. User starts a new report of another user.

2b. User cancels the report:
    1. The user selects “cancel report”.
    2. System closes the dialog box.

4a. User inputs invalid data (User name doesn’t exist, invalid character, empty text):
    1. System informs the user the input is invalid and prompts for the correct input.
    Repeat step 1 until the input is valid.

6a. User mistakenly reported other user:
    1. The user contacts admin.


sd Report User - Use Case
Rider/Driver
: System
1: fileReport(user name, reason)
1.1: confirmation of report

Use Case: Making Profile - Leighton Glim

Scope: Baylor Rideshare creating an account for drivers/riders

Level: User goal

Primary Actor: User

Stakeholders & Interests:

- User: Wants to create an account.

- System Administrators: Wants to allow account creation for Rider/Driver to offer/find rides and adds account to database.

Preconditions: The User has the software open.

Success Guarantee (Postconditions): The User has an account created.

Main Success Scenario (or Basic Flow):

1. The user selects “Create an Account.”
2. The System displays the account creation frame.
3. The user will enter their name, Baylor email address, phone number, expected graduation date and password (x2).
4. The System sends a verification email to the user’s email account.
5. The user receives the verification email and selects “account confirmation” in the email.
6. The System confirms the user’s account.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application and restart.
    2. The user creates a new profile if failure occurred during account creation.

3a. User has invalid data in the fields:
    1. The System prompts the user to input valid data.
    2. The user inputs data.
    Repeat step 1-2 until the input is valid.

4a. Verification email is not received:
    1. The user will select “back” from the verification screen.
    2. The System shows the account creation frame.
    3. The user updates their email.
    4. The System sends a new verification email to the entered address.

5a. User does not confirm account:
    1. The account is deactivated.

Use Case: Customize Profile - Leighton Glim

Scope: Baylor Rideshare drivers/riders customizing account

Level: User goal

Primary Actor: Users

Stakeholders & Interests:

- User: Wants to customize their account to more accurately fit their information

- System Administrators: Wants to allow account customization for users and changes account information in database.

Preconditions: The user has a valid account created, and the application is open.

Success Guarantee (Postconditions): The System records the updated account information.

Main Success Scenario (or Basic Flow):

1. The user selects “Customize Profile.”
2. The system displays the customization menu.
3. The user selects “Edit” on a field.
4. System prompts the user to enter new data that corresponds to the updated field.
5. The user enters new data.

   User repeats steps 3-5 until satisfied.

6. The user confirms their changes.
7. System saves changes and closes the editor frame.

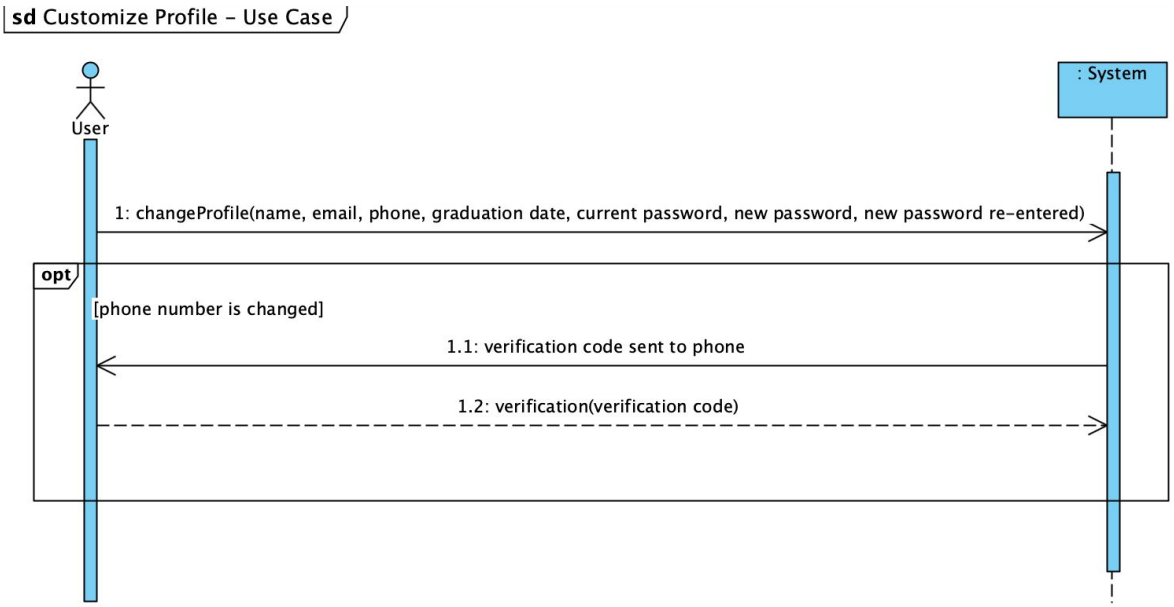
Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application and restart.
    2. The user selects “Customize Profile” to customize their profile.

3a. User accidently selects edit:
    1. User selects cancel
    2. System does not save the changes.

5a. The user enters invalid data:
    1. System prompts the user to input valid data.
    2. The user inputs data.
    Repeat step 1-2 until the input is valid.

6a. User cancels change
    1. Systems reverts fields back to initial values.



Use Case Generate Report - Andrew Ammentorp

Scope: Baylor Rideshare generating report

Level: User Goal

Primary Actor: Admin

Stakeholders & Interests:

- System Administrators: Want to view a record of activities in a given timeframe.

Preconditions: The system is functioning as intended.

Success Guarantee (Postconditions): The System records the report and delivers it to the admin.

Main Success Scenario (or Basic Flow):

1. Admin logs into the system.
2. System validates their login.
3. Admin selects “settings.”
4. System opens the settings menu.
5. Admin selects “generate report.”
6. System prompts admin to enter the start date and end date.
7. The admin enters the starting date, and the ending date, and selects “generate report”.
8. The System generates the report and presents it to the Admin.

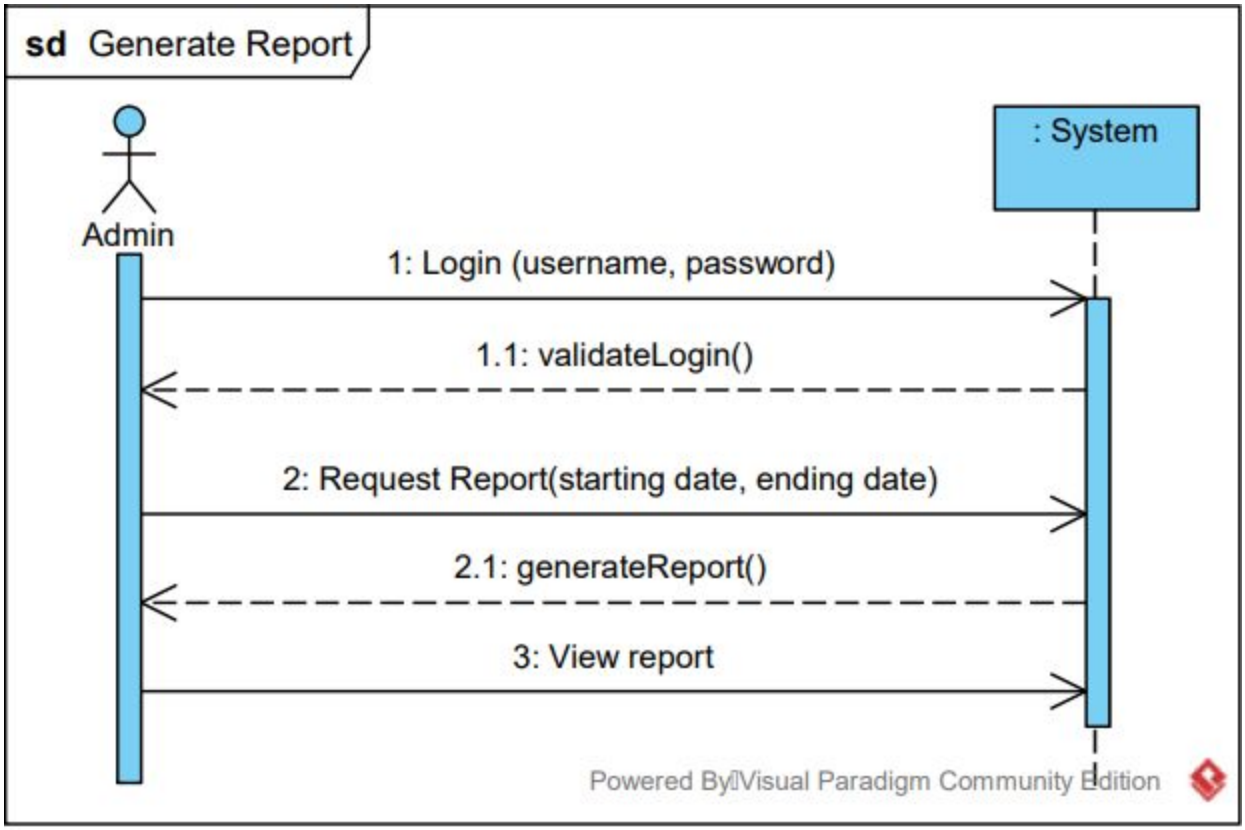
Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application and restart.
    2. The admin goes into setting and selects “generate report”.

1a. The login is unsuccessful:
    1.The admin re-enters their credentials correctly.
    2. System attempts to validate login again.
    Repeat step 1-2 until the login is successful.

3a. The input dates are invalid:
    1. The System shows an error, and prompts the Admin to enter a valid date range.
    2. The Admin enters a valid date range.

3b. The admin clicks cancel report:
    1.System clears page and goes back to main settings page



Use Case Update Post Database – Andrew Ammentorp

Scope: Baylor Rideshare post database

Level: Subfunction

Primary Actor: System operations

Stakeholders & Interests:

- System Operations: Wants to be able to update the posts on the app under certain conditions.

Preconditions: The System is functioning normally.

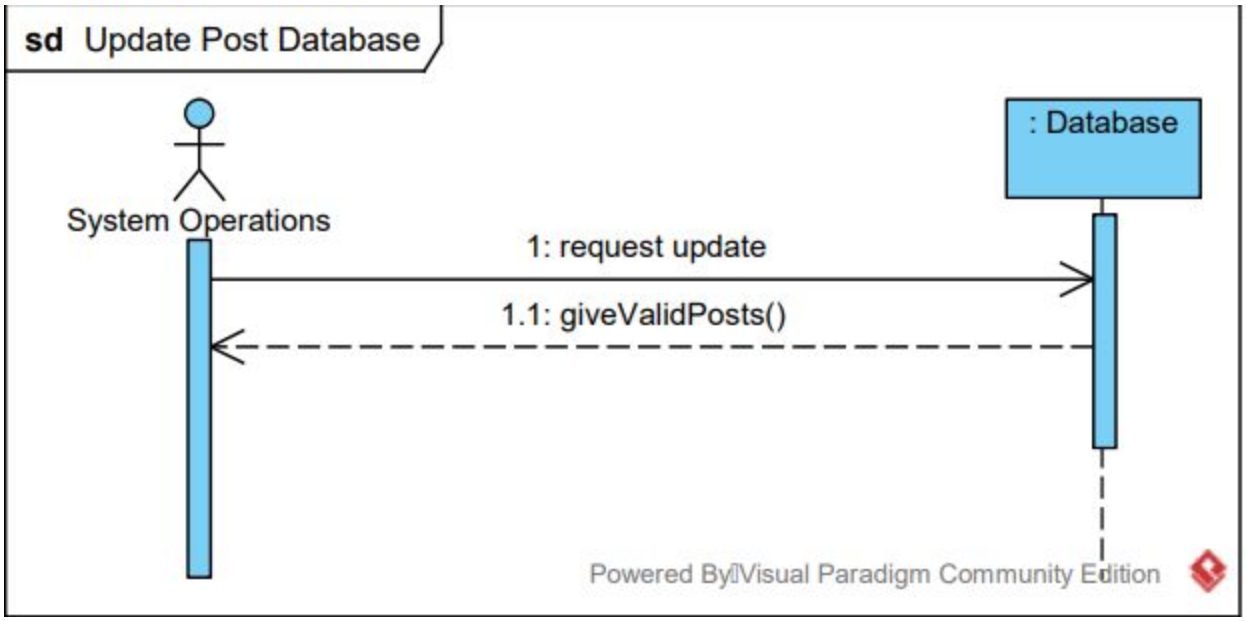
Success Guarantee (Postconditions): Rides that are past the date listed are removed from the database.

Main Success Scenario (or Basic Flow):

1. The system requests the database to update.
2. The database gives the valid posts to the system.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. System reboot.



Use Case Update User Database – Andrew Ammentorp

Scope: Baylor Rideshare user database

Level: Subfunction

Primary Actor: System Operations

Stakeholders & Interests:

- System Operations: Wants to be able to update the users under certain conditions.

Preconditions: The software has been created, with a number of profiles created by users.

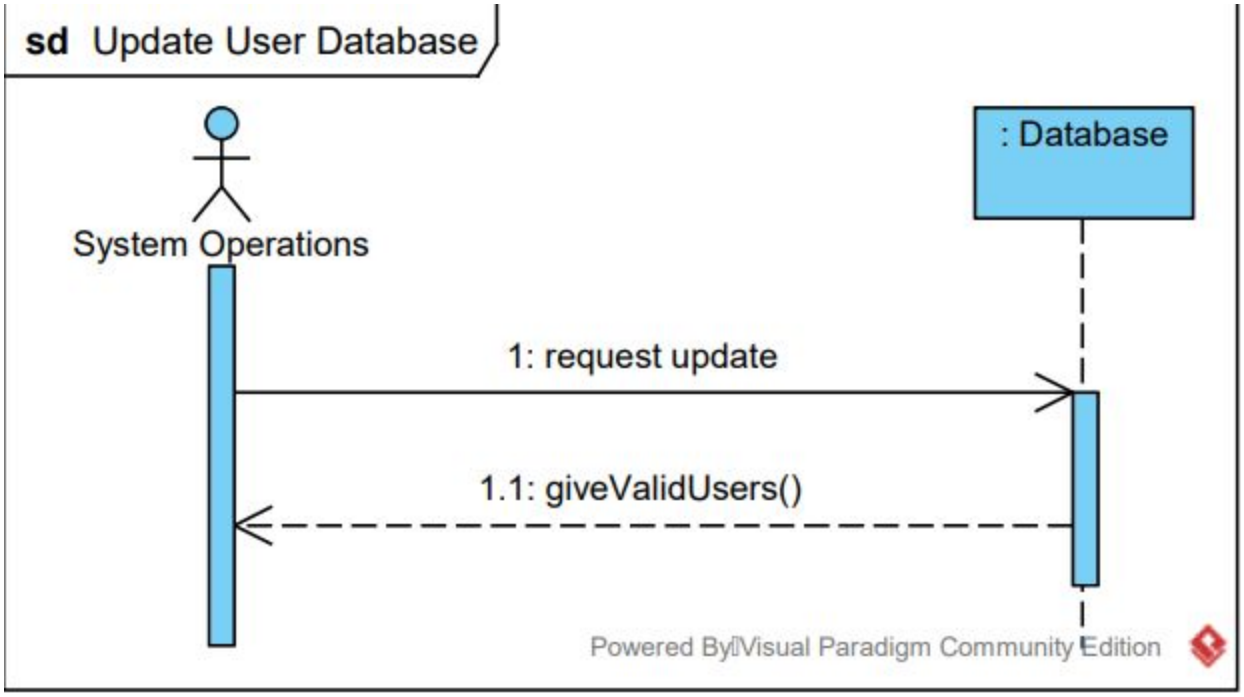
Success Guarantee (Postconditions): The system is functioning normally

Main Success Scenario (or Basic Flow):

1. The system requests the database to update.
2. The database gives the valid users to the system.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. System reboot.



Use Case: Search for Driver - Joshua Huertas

Scope: Baylor Rideshare finding a driver

Level: user goal

Primary Actor: Riders

Stakeholders & Interests:

- Rider: Wants to find an available driver easily.

- Driver: Wants to find available riders easily.

- System Administrators: Wants to record the ride shares and satisfy the needs of the users.

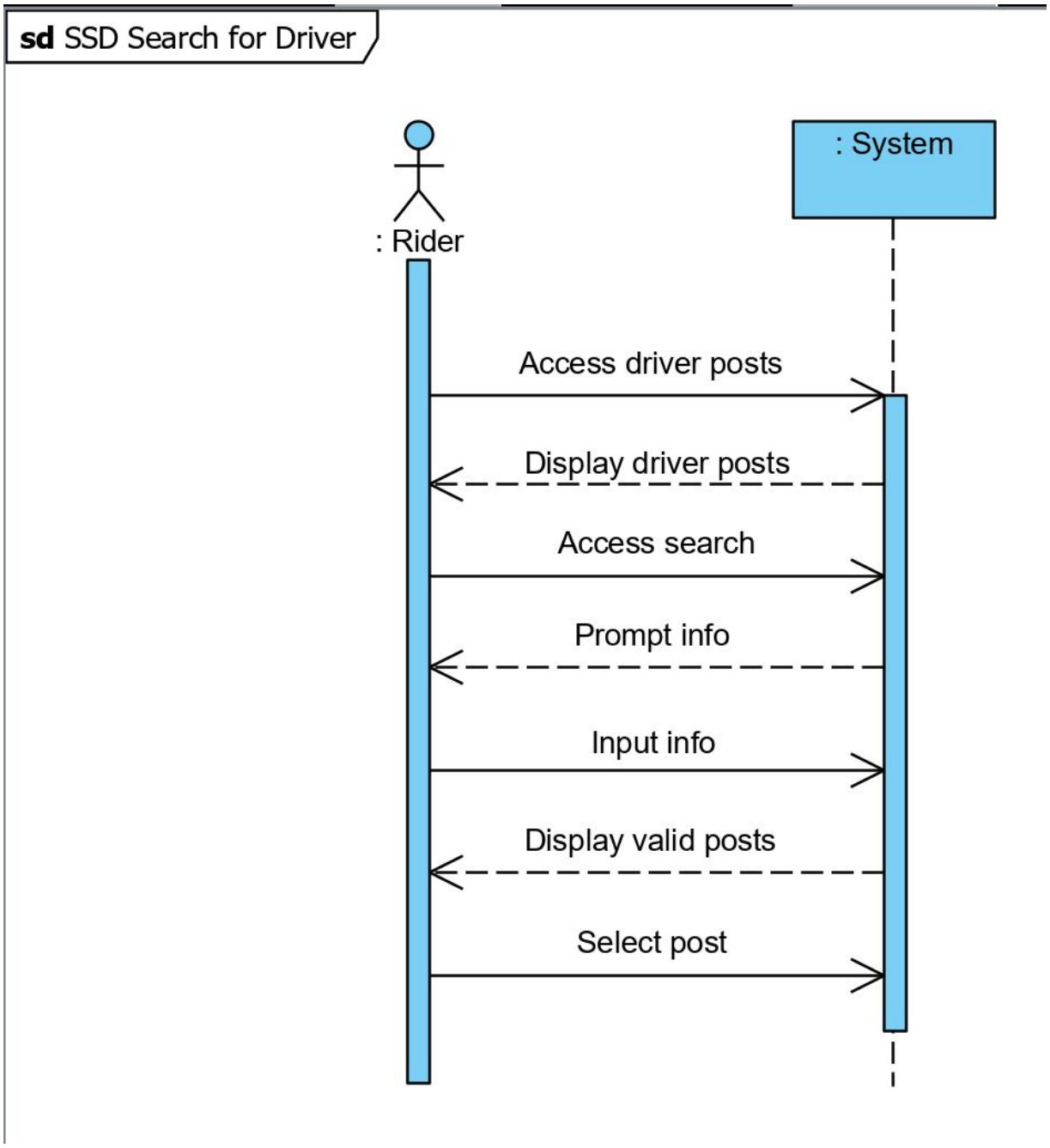
Preconditions: The Rider has registered an account on the Baylor Rideshare app.
Success Guarantee (Postconditions): The search is recorded by the system.

Main Success Scenario (or Basic Flow):

1. Rider selects driver posts.
2. System displays all active driver posts.
3. Rider selects the search bar.
4. The System prompts the rider for information, “point A, point B, and date.”
5. Rider inputs requested info.
6. System displays posts with matching information.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application.
    2. Restart the application.
5a. User has not filled all prompted fields:
    1. The System filters based on given fields.
6a. No matching posts exist:
    1. Display error message to screen
    2. The System prompts for another search



Use Case: Search for Rider - Joshua Huertas

Scope: Baylor Rideshare finding a driver

Level: user goal

Primary Actor: Drivers

Stakeholders & Interests:

- Driver: Wants to find available riders easily.

- Rider: Wants to find an available driver easily.

- System Administrators: Wants to record the ride shares and satisfy the needs of the users.

Preconditions: The Driver has registered an account on the Baylor Rideshare app.

Success Guarantee (Postconditions): The search is recorded by the system.

Main Success Scenario (or Basic Flow):

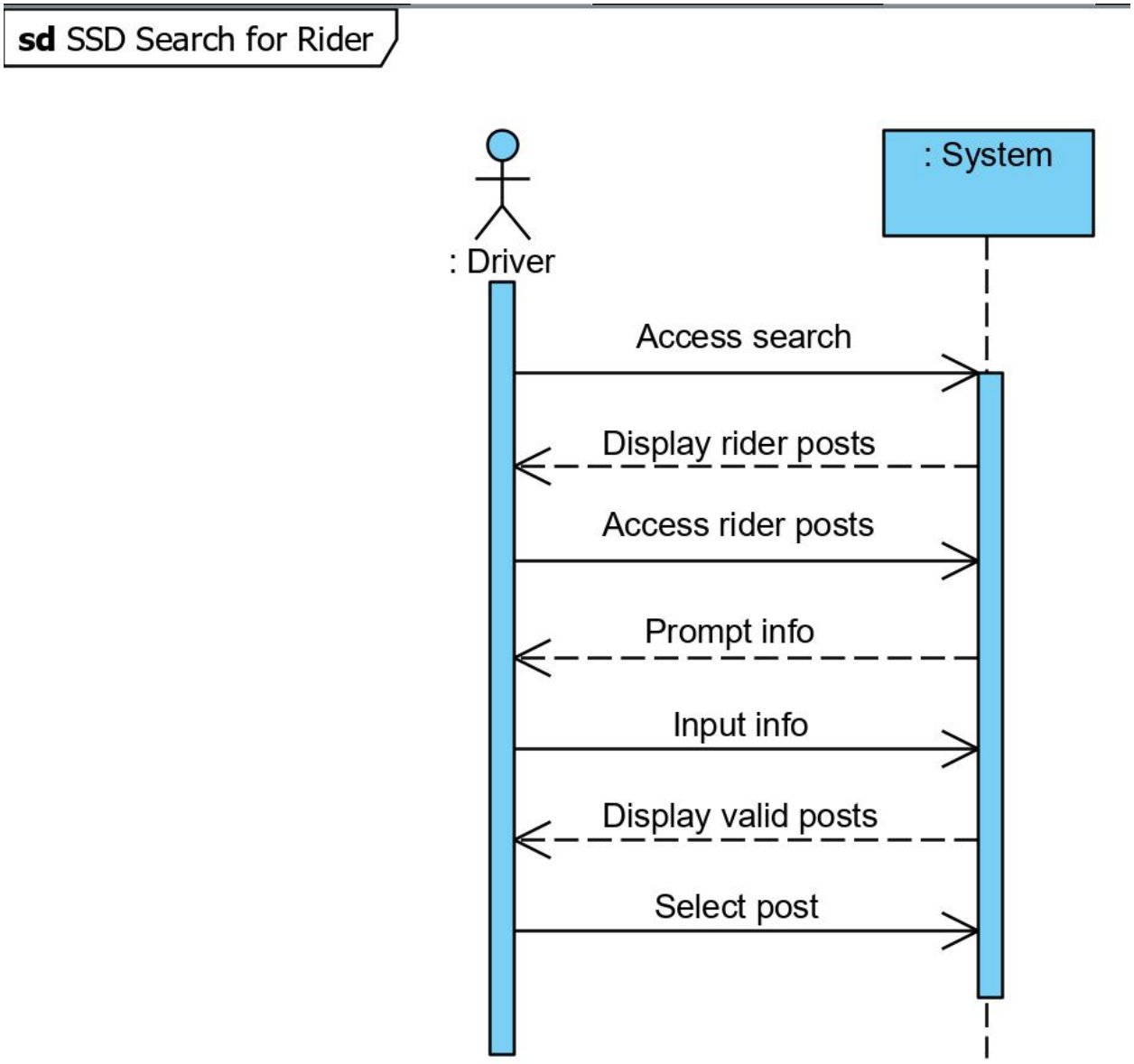
1. Driver selects rider posts.
2. System displays all active rider posts.
3. Driver selects the search bar.
4. The System prompts the driver for information, “point A, point B, and date.”
5. Driver inputs requested info.
6. System displays posts with matching information.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application.
    2. Restart the application.

5a. User has not filled all prompted fields:
    1. System filters based on given fields.

6a. No matching posts exist:
    1. Display error message to screen
    2. The System prompts for another search


sd SSD Search for Rider
: Driver
: System
Access search
Display rider posts
Access rider posts
Prompt info
Input info
Display valid posts
Select post

Use Case: Search database for post - Joshua Huertas

Scope: Baylor Rideshare finding a post

Level: Subfunction

Primary Actor: System

Stakeholders & Interests:

- User: Wants to find specific posts easily.

- System: Wants to search database to find relevant posts to satisfy user needs.

- System Administrators: Wants to find rideshares to satisfy the needs of the users.

Preconditions: The Database has been populated with posts and the user wants to search for posts with specific attributes.

Success Guarantee (Postconditions): The Database is unchanged and the desired posts are displayed to the user.

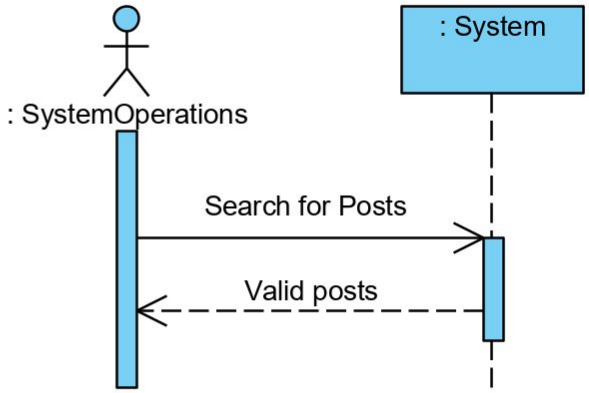
Main Success Scenario (or Basic Flow):

1. System requests to find active posts that match search requirements in the database.
2. Database returns the list of all matching active posts to the System.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. System reboot
2a. No matching posts exist:
    1. Set returns empty

sd SSD Search database for post



Use case: Survey User - Joseph Perez

Scope: Survey

Level: User Goal

Primary Actor: User

Stakeholders & Interests:

- User: wants to make sure their opinion is heard
- Admins: will use data to improve quality of user experience

Preconditions: The user has just arrived at their destination successfully.

Success Guarantee (Postconditions): The survey results are recorded.

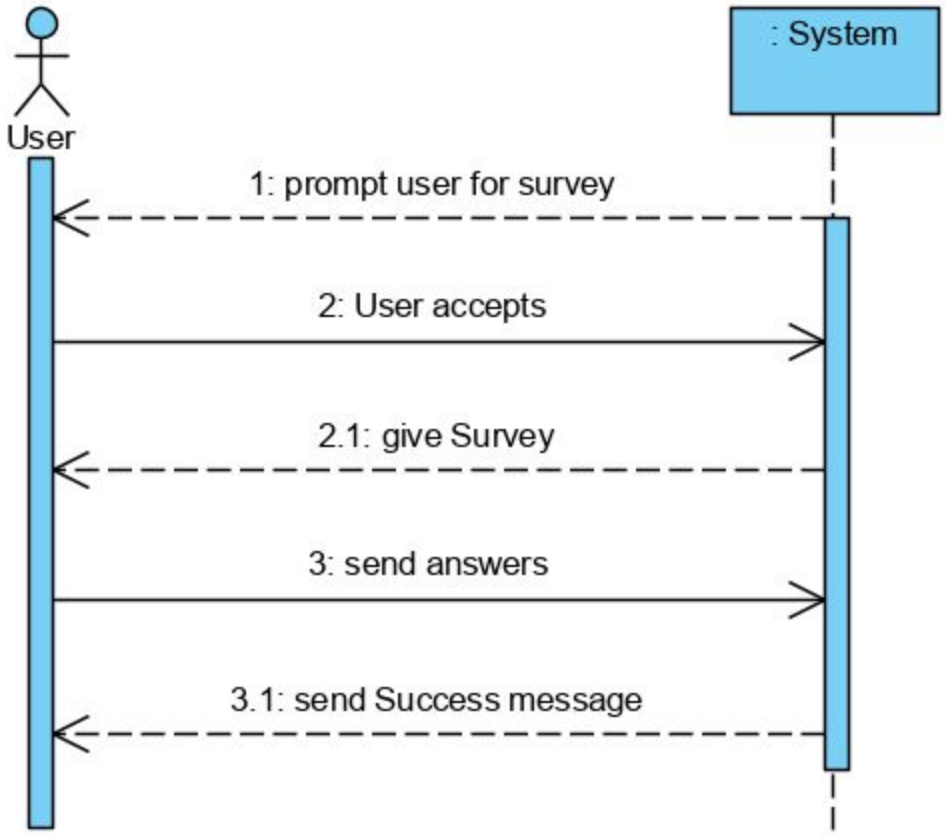
Main Success Scenario (or Basic Flow):

1. The System asks the user if they would like to answer a survey.
2. User agrees to take the survey.
3. System displays the window for the survey.
4. User answers the survey and sends responses.
5. System records the answers, sends a success message to the user and closes the survey.

Extensions (or Alternative Flows):

a* The user chooses to no longer participate in the survey:

1. The window closes.
2. System records the survey.


User
: System
1: prompt user for survey
2: User accepts
2.1: give Survey
3: send answers
3.1: send Success message

Use case: Confirming Post - Joseph Perez

Scope: Posting

Level: User goal

Primary Actor: User

Stakeholders & Interests:

- Users want to share contact information only with other users they will be riding with.

Preconditions:

There must be an open post.

Success Guarantee (Postconditions):

Rider's request is processed by driver

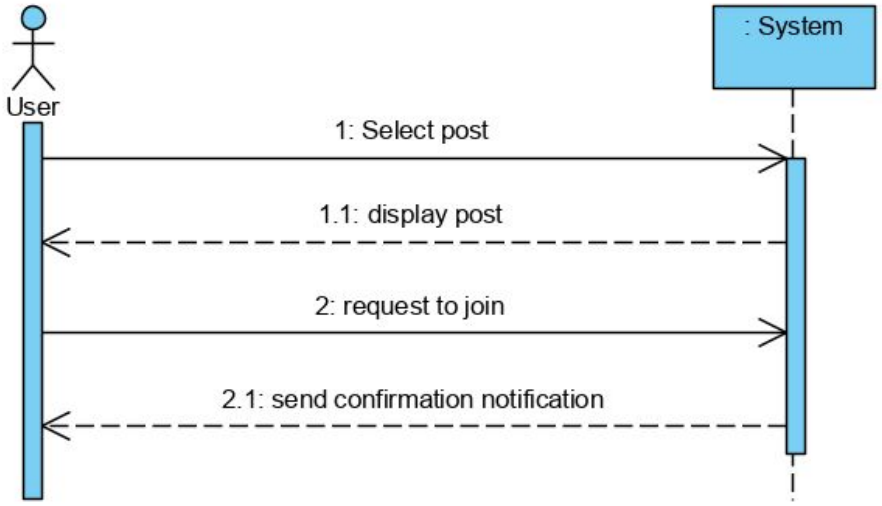
Main Success Scenario (or Basic Flow):

1. User selects a post.
2. The system displays the post.
3. User selects request to join the ride
4. System sends notification to poster
5. User is notified of request's status

Extensions (or Alternative Flows):

*a. At any time, System fails:

2. System reboot



Use case: Remove Inappropriate posts - Joseph Perez

Scope: Inappropriate Behavior

Level: Summary

Primary Actor: Admin

Stakeholders & Interests:

- Users: do not want to view inappropriate behavior
- Admins: want to ensure posts do not contain inappropriate content

Preconditions: A post is reported to the Admin.

Success Guarantee (Postconditions):

Issue is resolved by Admin.

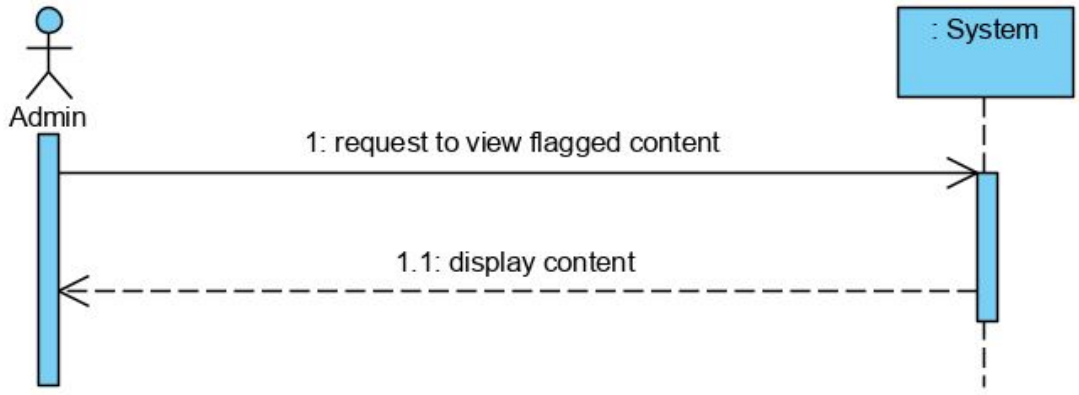
Main Success Scenario (or Basic Flow):

1. Admin requests to review flagged content.
2. System displays flagged content.
3. Admin reviews content and takes appropriate steps to resolve the issue.

Extensions (or Alternative Flows):

*a. At any time, System fails:

1. Close the application.
2. Restart the application.



Use Case: Delete Account - Mohsen Soltani

Scope: Managing User Account

Level: User Goal

Primary Actor: User

Stakeholders & Interests:

- User: Wants to remove information from application easily and thoroughly
- Admin: Wants accurate record of current users

Preconditions: User has an existing account.

Success Guarantee (Postconditions): Post Database updates.

Main Success Scenario (or Basic Flow):

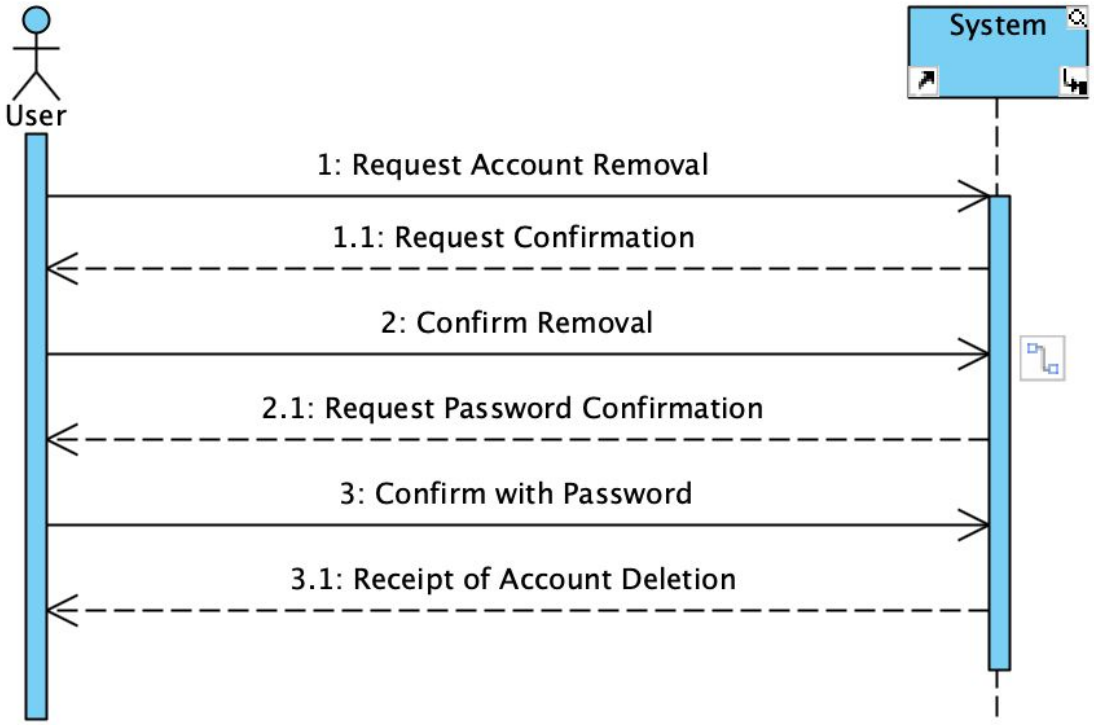
1. User selects “Delete Account” on own profile
2. System requests additional confirmation
3. User enters password
4. System removes the user account from the database, returns receipt of account removal.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application.
    2. Restart the application.

1.a: User doesn’t select “Yes”
    1. System aborts operation, user is returned to account window.

3.a: User enters incorrect password
    1.System aborts operation, user is returned to account window


User
System
1: Request Account Removal
1.1: Request Confirmation
2: Confirm Removal
2.1: Request Password Confirmation
3: Confirm with Password
3.1: Receipt of Account Deletion

Use Case: Remove Post - Mohsen Soltani

Scope: Managing User Account

Level: User Goal

Primary Actor: User

Stakeholders & Interests:

- User: Wants to remove post easily
- System: Wants to provide accurate rides to users.
- Database: Wants accurate record of posts

Preconditions: User post exists in database

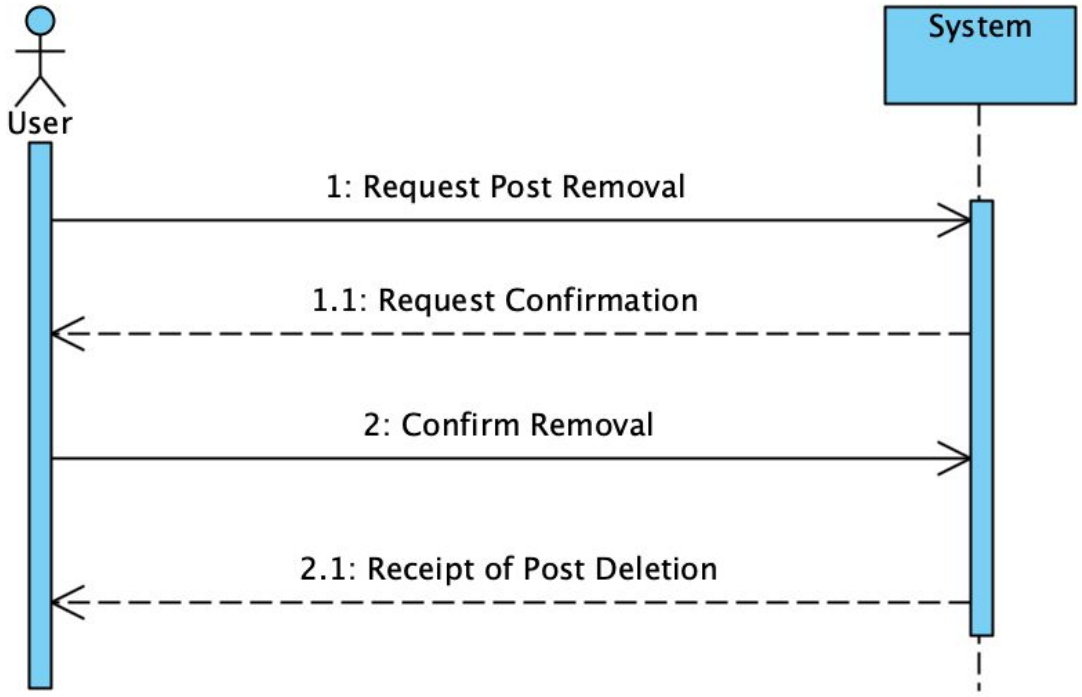
Success Guarantee (Postconditions): Post Database updates.

Main Success Scenario (or Basic Flow):

1. The User selects “Delete Post” on one of their own posts.
2. System requests confirmation.
3. User confirms post removal.
4. System removes the post from the database.
5. System returns a receipt of removal.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application.
    2. Restart the application.
3.a: User doesn’t confirm
    1. System aborts operation, returns user to post feed.


User
System
1: Request Post Removal
1.1: Request Confirmation
2: Confirm Removal
2.1: Receipt of Post Deletion

Use Case: Suspend User - Mohsen Soltani

Scope: Admin Management

Level: Admin Goal

Primary Actor: Admin

Stakeholders & Interests:

- Admin: Wants to suspend users and manage them easily
- Database: Wants accurate record of suspended users
- System: Notifies users of suspension

Preconditions: User is not suspended

Success Guarantee (Postconditions): User Database is updated.

Main Success Scenario (or Basic Flow):

1. Admin chooses user to suspend, based on potential infraction.
2. System returns account.
3. Admin reports reason for suspension.
4. System prompts admin for suspension confirmation.
5. Admin confirms user suspension.
6. System revokes user permissions.
7. Database marks user as suspended.
8. System alerts user, and gives suspension receipt to user.

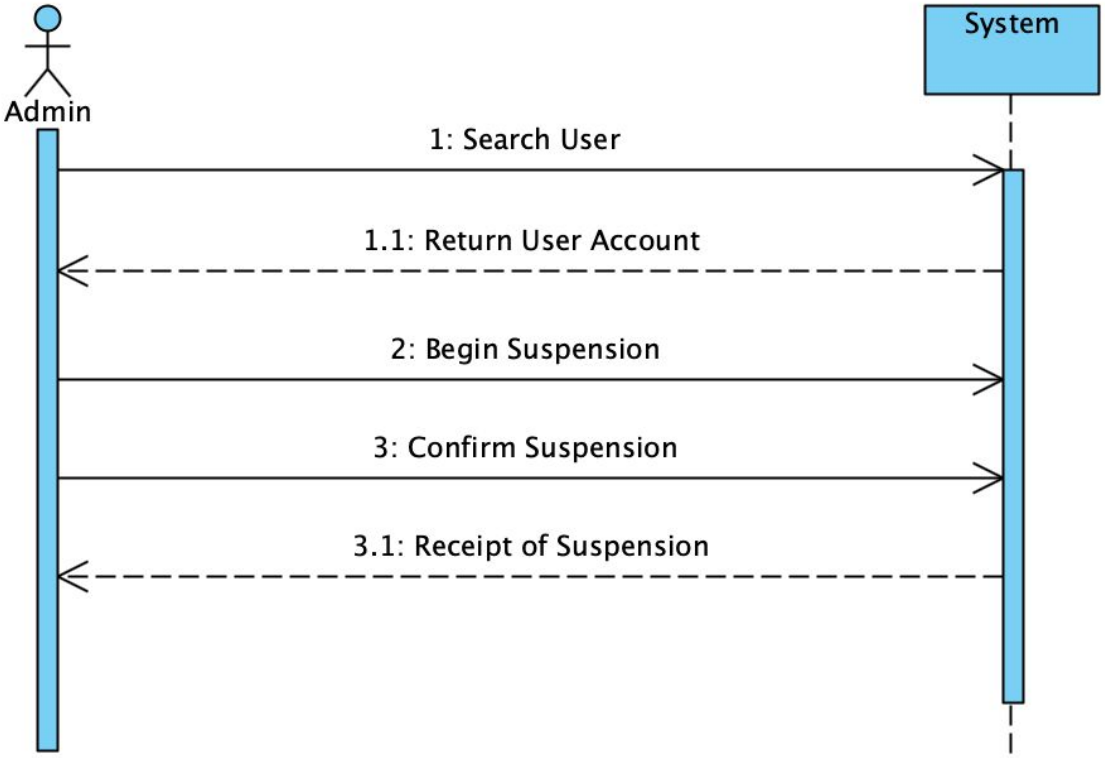
Extensions (or Alternative Flows):

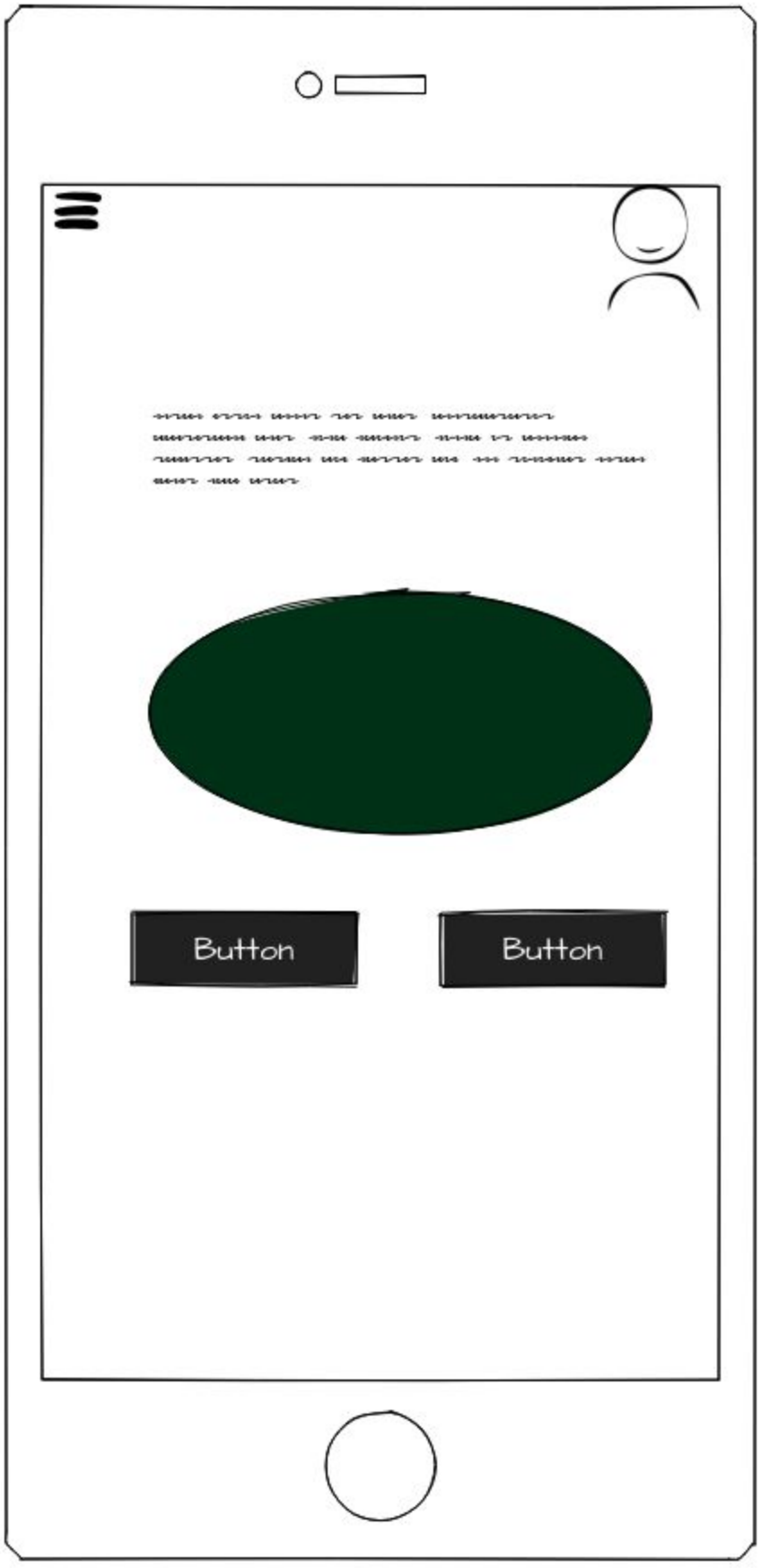
*a. At any time, System fails:
    1. Close the application.
    2. Restart the application.

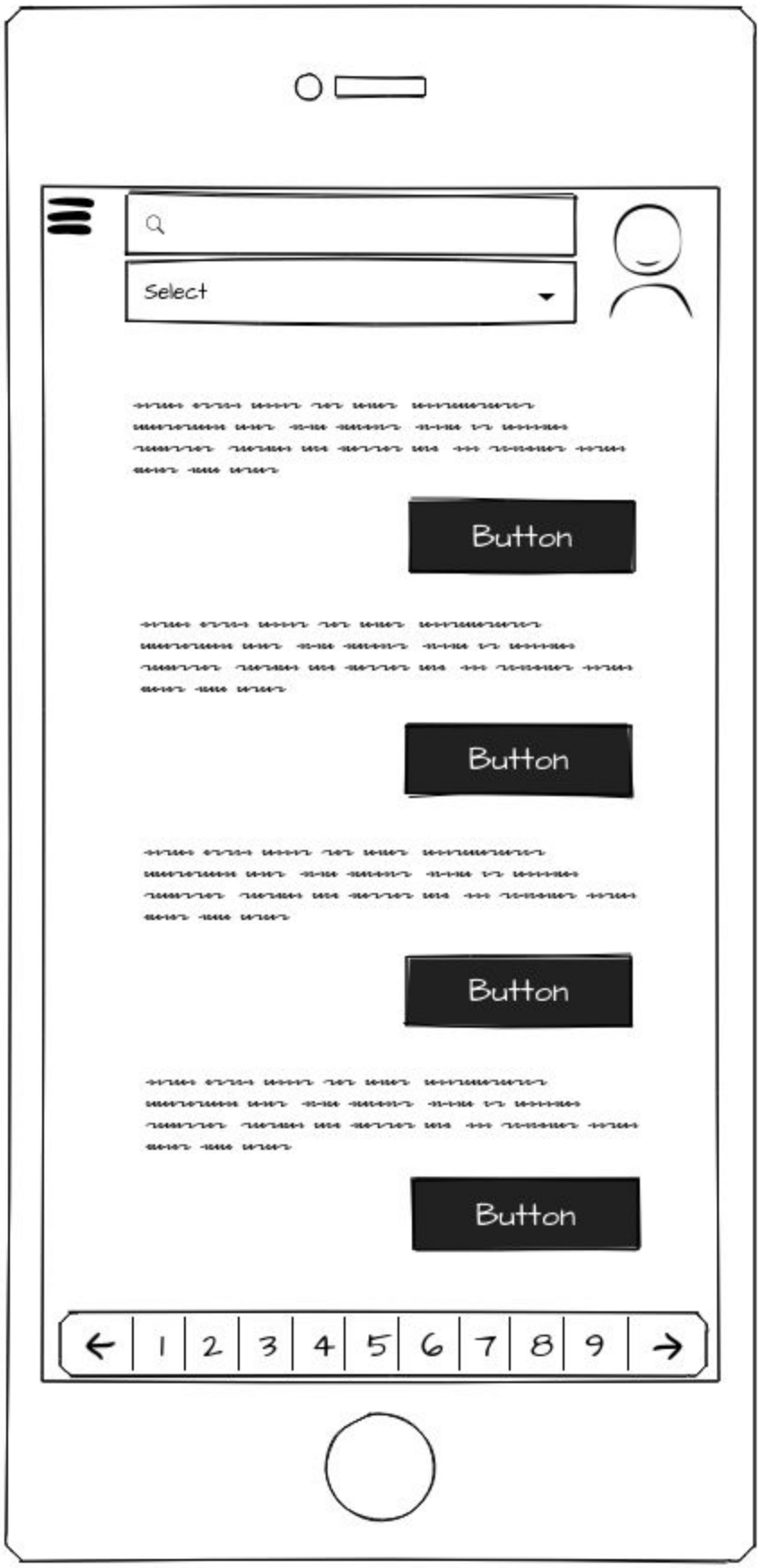
1.a: User cannot be found
    1. System alerts admin
    2. Admin restarts process

3.a: Reason left empty
    1. System prompts admin to provide reason for suspension
    Repeat until reason is given

4.a: Admin does not confirm
    1. System aborts operation, admin returns to user search


Admin
System
1: Search User
1.1: Return User Account
2: Begin Suspension
3: Confirm Suspension
3.1: Receipt of Suspension

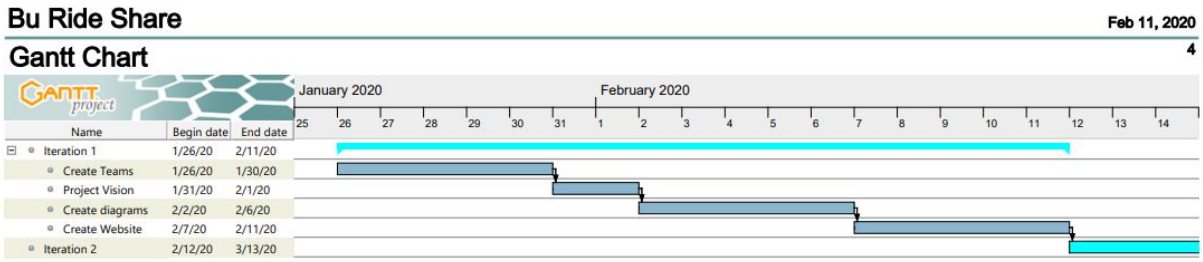

Button
Button

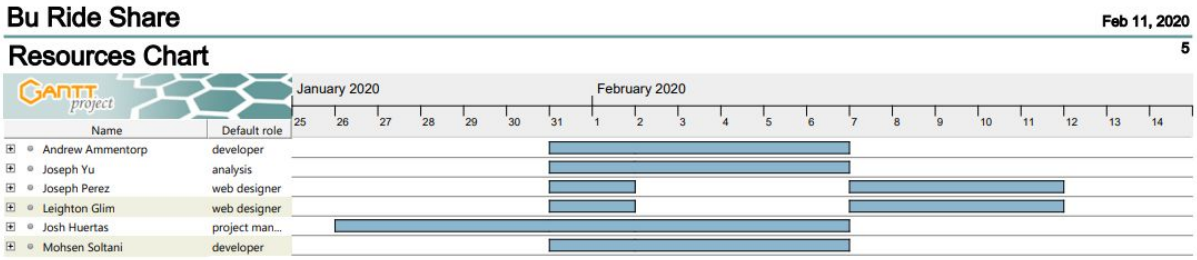

Select
Button
Button
Button
Button
1
2
3
4
5
6
7
8
9

https://burs-co.github.io/CSI3471-Ride-Share/ - website

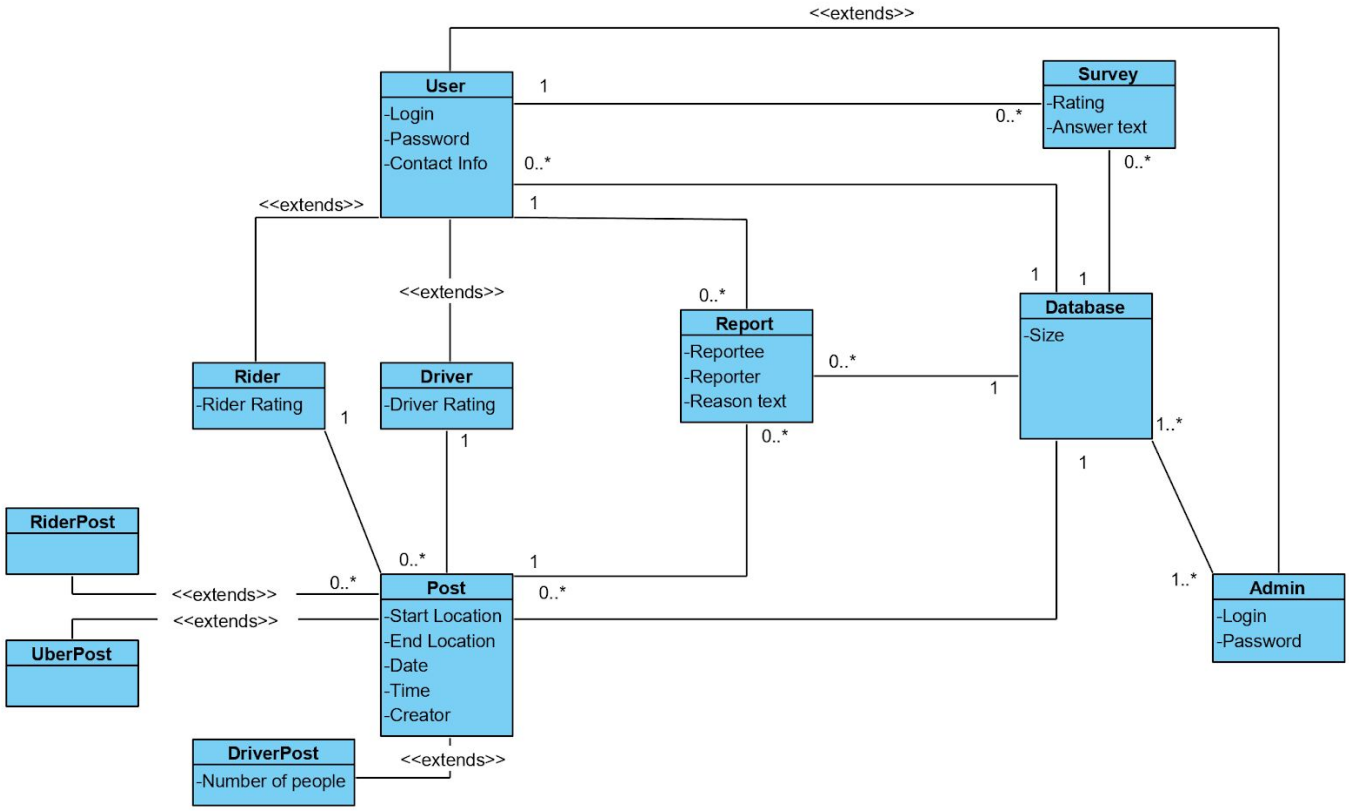
https://github.com/BURS-co/CSI3471-Ride-Share - git repository

Gantt





Domain Model



Traceability Matrix

Baylor Ride Share	Make User Profile	Verify w/BU EMail	App Tutorial	Sep driver/rider posts	Create new driver post	Create new rider post	Filter posts by date/loc	Sort posts by time	Upon, new post show suggestions for users who meet requirements	Remove expired rides	User can cancel posts	Remove canceled posts from feed	Rider/Driver must both confirm	Remove 100% confirmed rides	Waitlist riders if not 100% confirmed	Share contact info when rides are confirmed	Push notifications	Set reminders for Users
Make Rider Post				x		x			x				x				x	x
Make Driver Post				x	x				x				x				x	x
Make Uber Post				x		x			x				x			x	x	x
Report User	x																	
Search for Driver							x	x					x	x	x	x		
Search for Riders							x	x					x	x	x	x		
Delete Account	x		x															
Remove Post					x	x				x	x							
Make Profile	x	x	x															
Survey User	x																	
Confirming Post					x	x												
Customize Profile	x	x	x															
Suspend User	x	x																
Remove Inappropriate Posts					x							x						
Search Database for post							x	x	x	x	x	x	x	x	x	x		
Generate Report	x	x		x	x	x	x								x	x		
Update Post Database										x	x	x		x	x			
Update User Database	x																x	x