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.

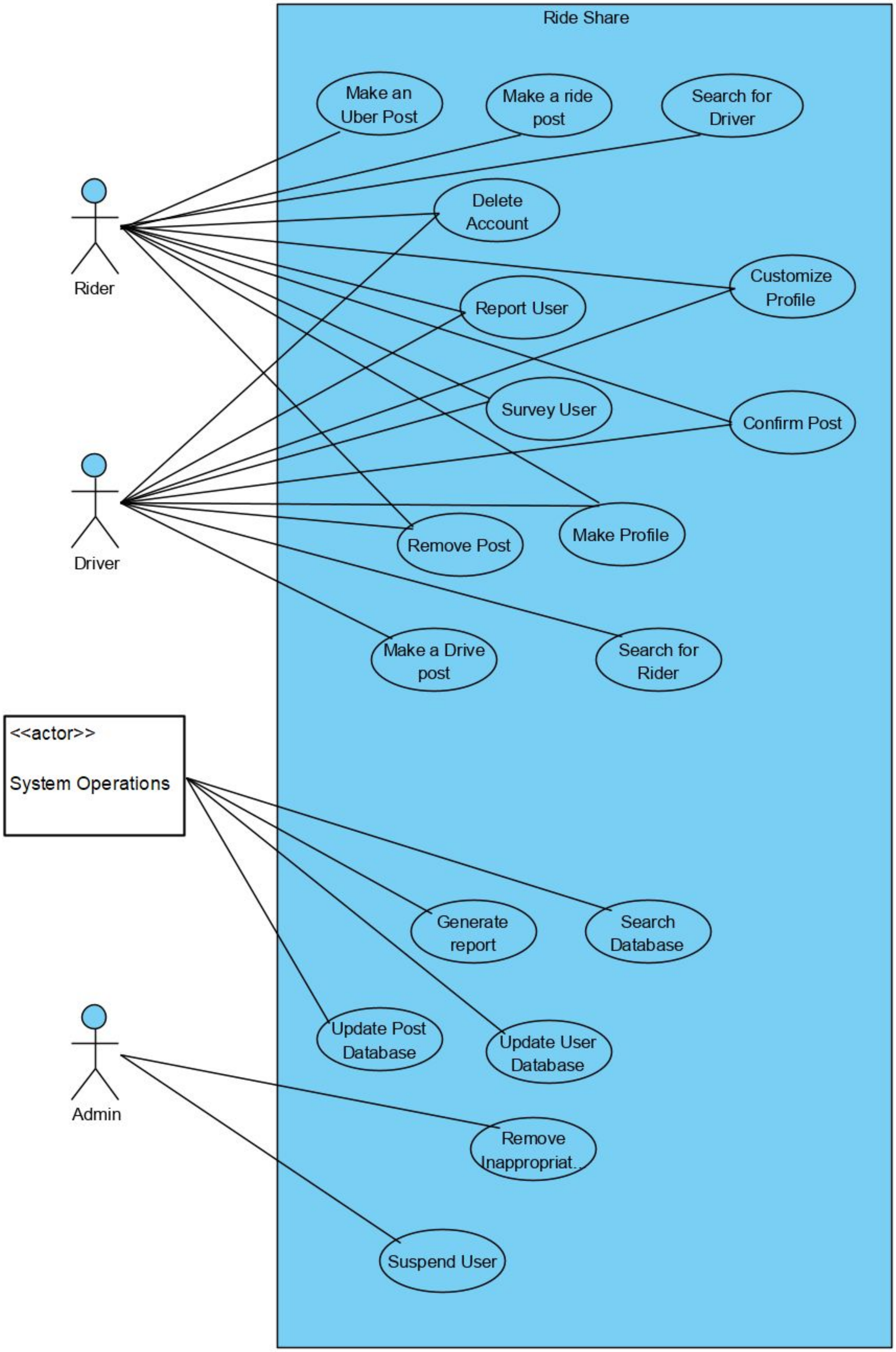
UseCases + System Sequence Diagram:

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
Admin
Remove Inappropriat...
Suspend User

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 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 ride is recorded by the system.

Main Success Scenario (or Basic Flow):

1. The rider requires transportation from point A to point B.
2. The rider opens the application and clicks on “new ride post.”
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 a request to the Driver.
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.
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 abandons the request:

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

5a. The Rider cancels the new Ride Post:

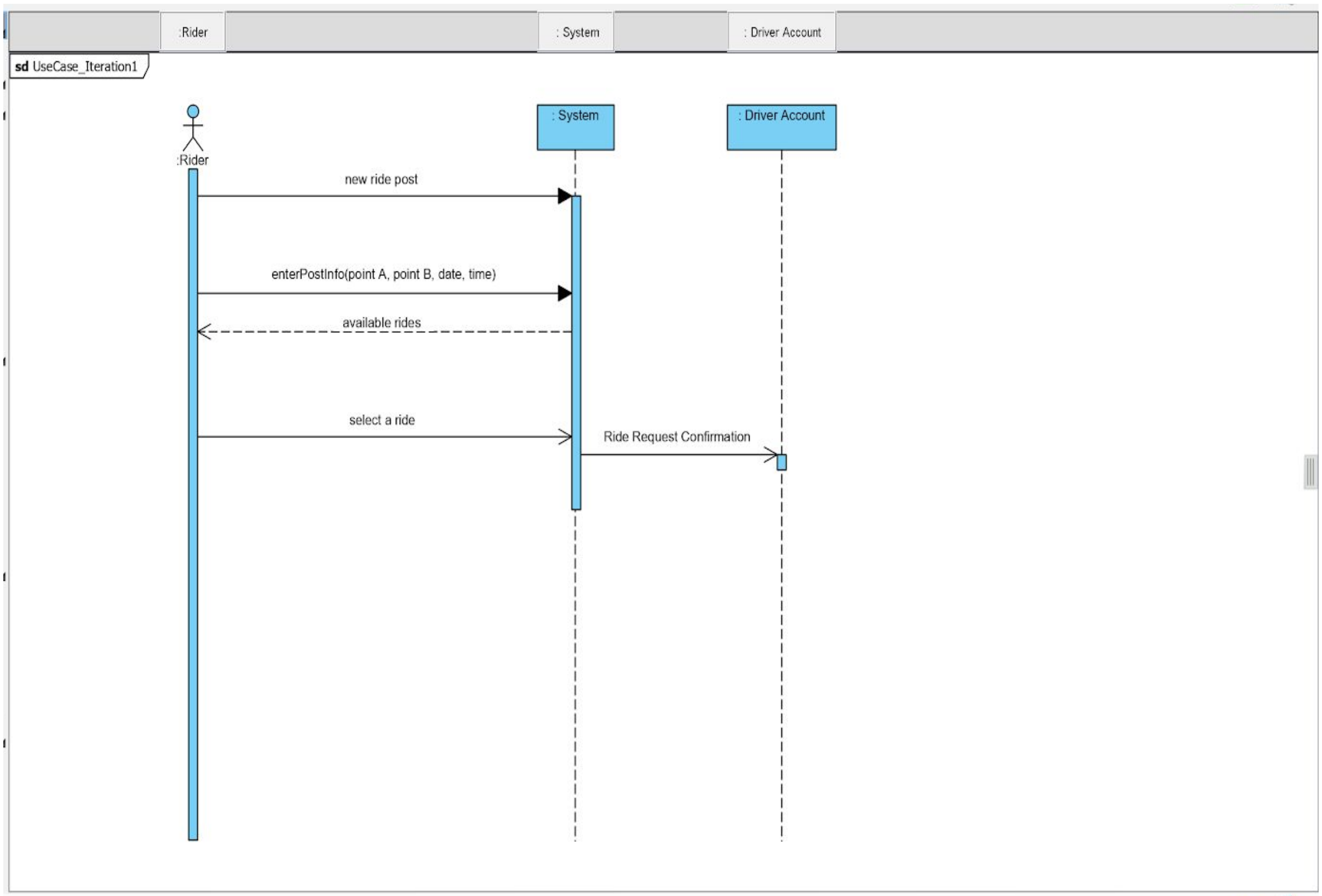
1. The System records the cancelled request, the rider request is not posted.

5b. The Rider is on the current page too long without making a selection:

1. The System notifies the Rider that the page have been idle for too long and ask for the Rider’s confirmation.
2. The System runs the query again and updates the search results.

2a. The Rider fails to confirm after a set amount of time:

1. The System records the request, and the rider request is cancelled.



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: 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 ride is recorded by the system.

Main Success Scenario (or Basic Flow):

1. The driver wants to provide a ride for potential riders.
2. The driver opens the application and clicks on "new driver post."
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).
7. The Rider closes the application.

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.
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:

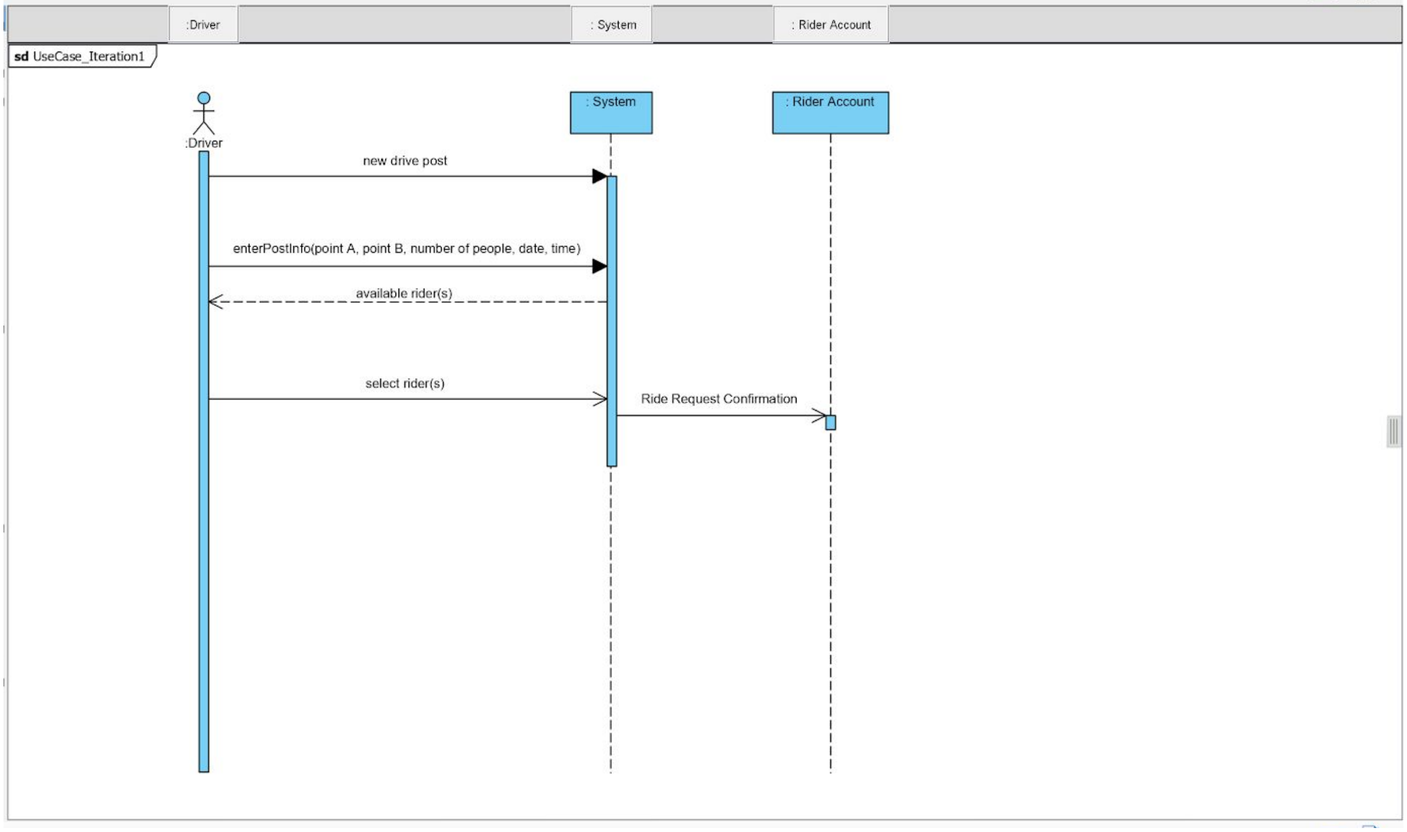
1. The System records the driver post offer, and the new driver post is not posted.

5b. The Driver is on the current page too long without making a selection:

1. The System notifies the Driver that the page have been idle for too long and ask for the Driver's confirmation.
2. The System runs the query again and updates the search results.

    2a. The Driver fails to confirm after a set amount of time:

    1. The System records the driver post offer, and the new Driver post is cancelled.



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 ride is recorded by the system.

Main Success Scenario (or Basic Flow):

1. The rider requires transportation from point A to point B.
2. The rider opens the application and clicks on "new Uber ride post."
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 a request 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.

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 abandons the request:

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

5a. The Rider cancels the new Ride Post:

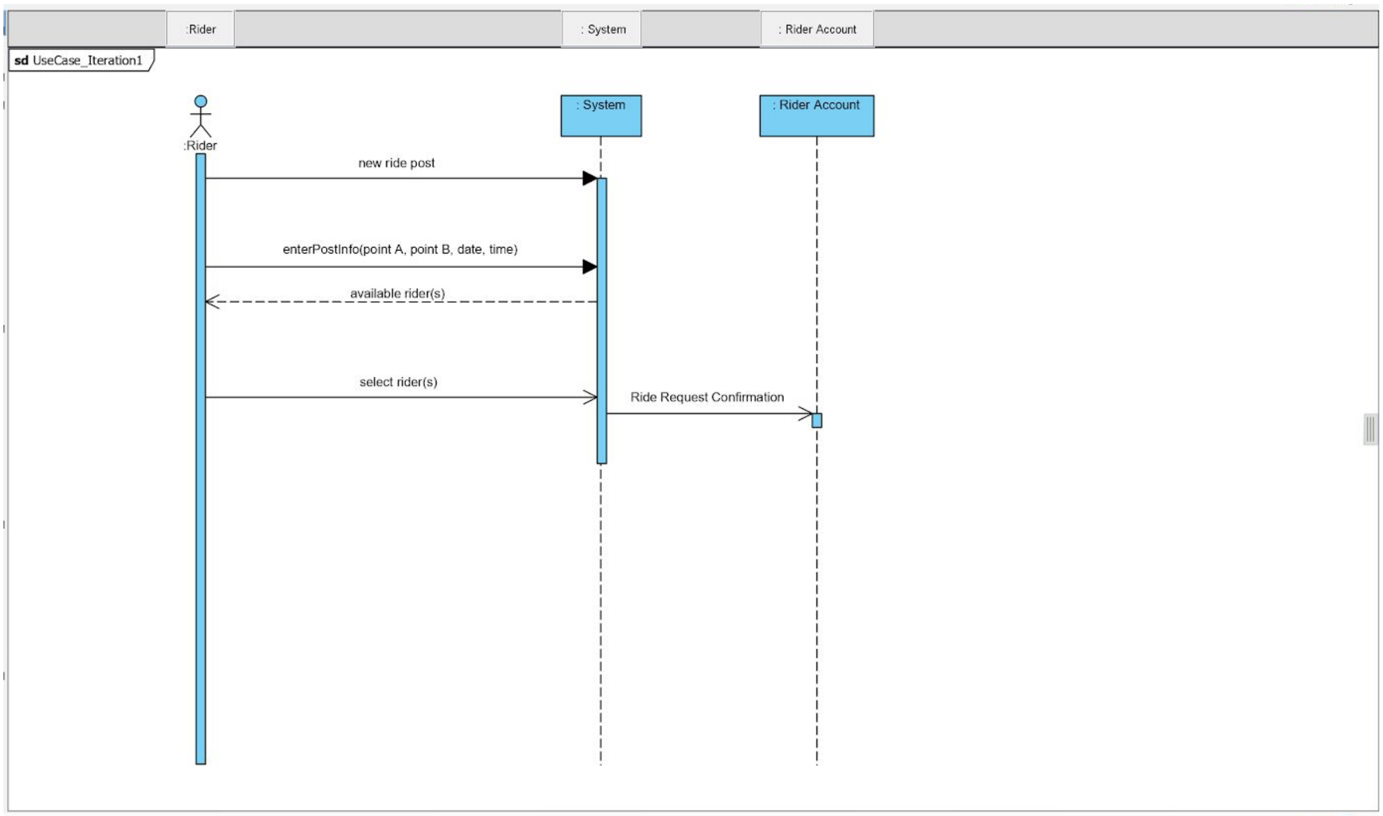
1. The System records the cancelled request, the rider request is not posted.

5b. The Rider is on the current page too long without making a selection:

1. The System notifies the Rider that the page have been idle for too long and ask for the Rider’s confirmation.
2. The System runs the query again and updates the search results.

2a. The Rider fails to confirm after a set amount of time:

1. The System records the request, and the rider request is cancelled.



Use Case: Report User - Leighton Glim

Scope: Baylor Rideshare reporting drivers/riders

Level: User goal

Primary Actor: Drivers and Riders

Stakeholders & Interests:

- Rider: Wants to report bad experience with driver or other users

- Driver: Wants to report bad experience with rider or 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 and appropriate actions are taken by System Administrator.

Main Success Scenario (or Basic Flow):

1. User (Driver or Rider) partakes in ride
2. The user opens the app and clicks “Report User.”
3. The user enters “name of user to report, reason for reporting.”
4. The application shows a summary of the report after confirming the report.
5. The user closes the application.
6. The System Administrator takes appropriate actions on the reported user.

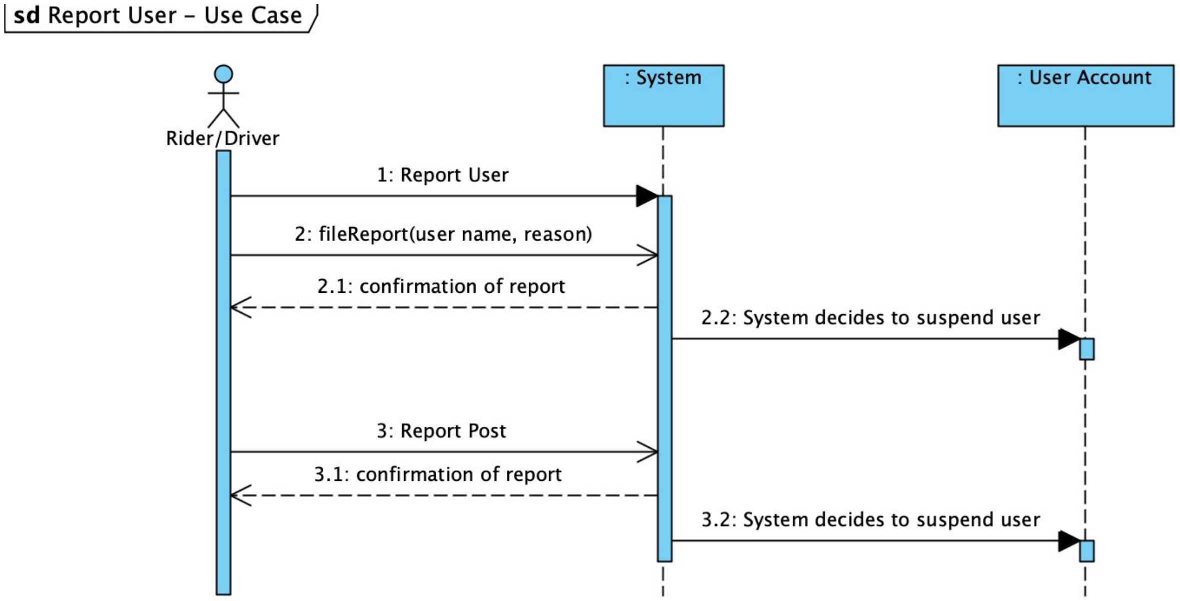
Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application and restart.
    2. User starts a new report of another user.
1a. User views inappropriate or unsafe content on the app:
    1. On post with inappropriate or unsafe content, user selects “Report Post”.
    2. Confirmation that the user wants to report post shows.
    3. Post is removed.
    4. The System Administrator takes appropriate actions on the user with post reported.
4a. User mistakenly reported other user or post:
    1. User views “My Reports”.
    2. User cancels report.

3. Actions previously taken on user or post from that report are reversed by System Administrator

4b. User is on page confirming action to report user or post for too long:

1. The System notifies the user that the page has timed out.
2. User starts a new report.


sd Report User – Use Case
Rider/Driver
: System
: User Account
1: Report User
2: fileReport(user name, reason)
2.1: confirmation of report
2.2: System decides to suspend user
3: Report Post
3.1: confirmation of report
3.2: System decides to suspend user

Use Case: Making Profile - Leighton Glim

Scope: Baylor Rideshare creating an account for drivers/riders

Level: User goal

Primary Actor: Drivers and Riders

Stakeholders & Interests:

- Rider: Wants to make an account to find future rides

- Driver: Wants to make an account to find future rides to offer

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

Preconditions: The Driver/Rider has downloaded and installed the Baylor Rideshare app.

Success Guarantee (Postconditions): The Driver/Rider has an account created.

Main Success Scenario (or Basic Flow):

1. User (Driver or Rider) opens the Baylor Rideshare app.
2. The user clicks “Create an Account.”
3. The user enters “name, Baylor email address, phone number, expected graduation date, password (x2).”
4. The user receives a verification code texted to their phone number.
5. The user enters the verification code and can now fully use the app.

Extensions (or Alternative Flows):

*a. At any time, System fails:
  1. Close the application and restart.
  2. User makes a new profile if failure occurred during process.

3-4a. User incorrectly enters email address:
  1. Close the application and restart.
  2. User clicks “Create an Account.”
  3. User enters correct Baylor email address
  4. User enters verification code sent to their phone

3-4b. Email address entered is not a Baylor email address:
  1. User is notified during profile creation process to enter valid email address

3-4c. Baylor email address is already taken:
  1. User selects “Report Issue.”

4a. Verification code is not received (invalid phone number):
  1. User clicks the back button from the verification screen.

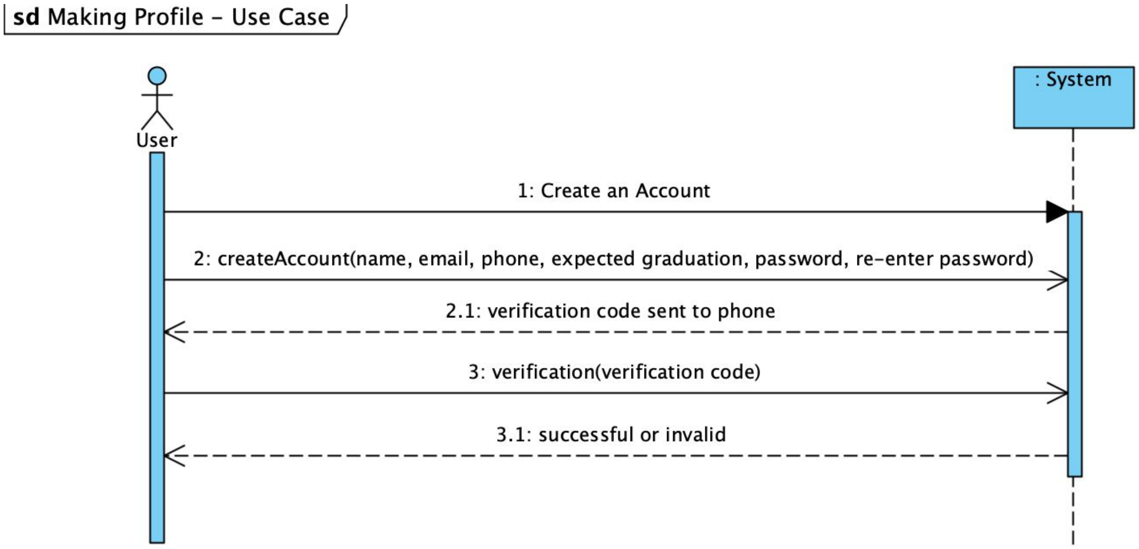
    2. User changes the phone number.
    3. Verification code is text to entered phone number.

4b. Verification code is not received:

    1. User selects “Send new verification code.”
    2. User enters new verification code texted to their phone number.

5b. Verification code entered is invalid:

    1. User selects “Send new verification code.”
    2. User enters new verification code texted to their phone number.


sd Making Profile - Use Case
User
: System
1: Create an Account
2: createAccount(name, email, phone, expected graduation, password, re-enter password)
2.1: verification code sent to phone
3: verification(verification code)
3.1: successful or invalid

Use Case: Customize Profile - Leighton Glim

Scope: Baylor Rideshare drivers/riders customizing account

Level: User goal

Primary Actor: Drivers and Riders

Stakeholders & Interests:

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

- Driver: 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.

Success Guarantee (Postconditions): The user has customized their account.

Main Success Scenario (or Basic Flow):

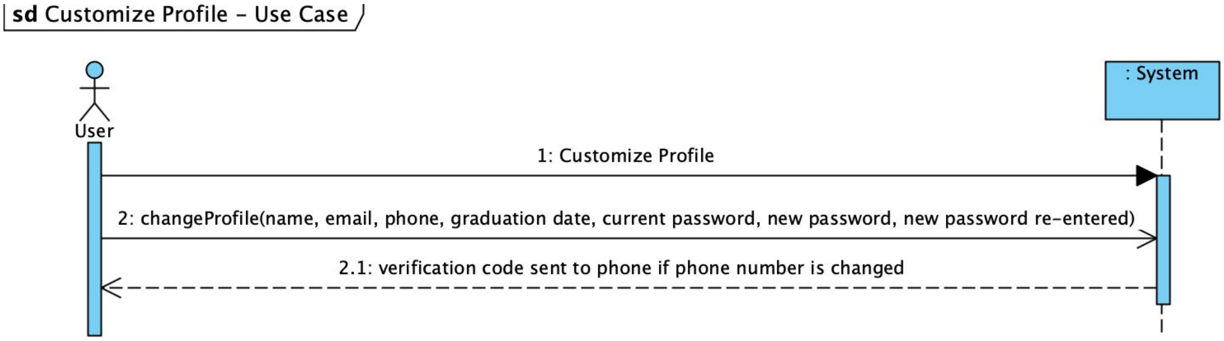
1. User (Driver or Rider) opens the Baylor Rideshare app.
2. The user clicks “Customize Profile.”
3. The user selects “Edit.”
4. The user changes name, email, profile picture, phone number, graduation date, or password.
5. The user selects “Save Profile”.

Extensions (or Alternative Flows):

*a. At any time, System fails:
    1. Close the application and restart.
    2. User clicks “Customize Profile”.
    3. User changes what they choose to alter on their profile.
1. Graduation date is approaching:
    1. User changes graduation date if is different from before.
4a. Changed email is already in use:
    1. User clicks “Report Issue.”
    2. User enters that their Baylor email address is already in use and account was not made by them.
4b. Email entered is not a Baylor email address:
    1. User is notified to enter a Baylor email address:
4b. Phone number is changed:
    1. Verification code is sent to the new phone number.
    2. User enters verification code.

4c. Password is to be changed:

1. User enters their current password.
2. User enters a new password.
3. User re-enters new password


sd Customize Profile - Use Case
User
: System
1: Customize Profile
2: changeProfile(name, email, phone, graduation date, current password, new password, new password re-entered)
2.1: verification code sent to phone if phone number is changed

Use Case Generate Report - Andrew Ammentorp

Scope: Baylor Rideshare generating report

Level: Subfunction

Primary Actor: Admin

Stakeholders & Interests:

- System Administrators: Wants to view a record of rides confirmed, cancelled, user accounts created, deleted, suspended/removed in a given timeframe.

Preconditions: The software has been created, and at least 1 day has gone by

Success Guarantee (Postconditions): A report has been given to the admin, listing a record of rides confirmed, rides cancelled, accounts created, and accounts deleted/suspended in the given timeframe.

Main Success Scenario (or Basic Flow):

1. Admin logs on to the website, using their administrator credentials.
2. Admin goes into setting, and clicks generate report
3. The admin specifies the starting date, and the ending date, and clicks generate report
4. All rides confirmed, rides cancelled, accounts created, and accounts deleted/suspended that occurred within the given timeframe are printed in a new window for the Admin to view at their pleasure.

Extensions (or Alternative Flows):

1a. The login is unsuccessful.

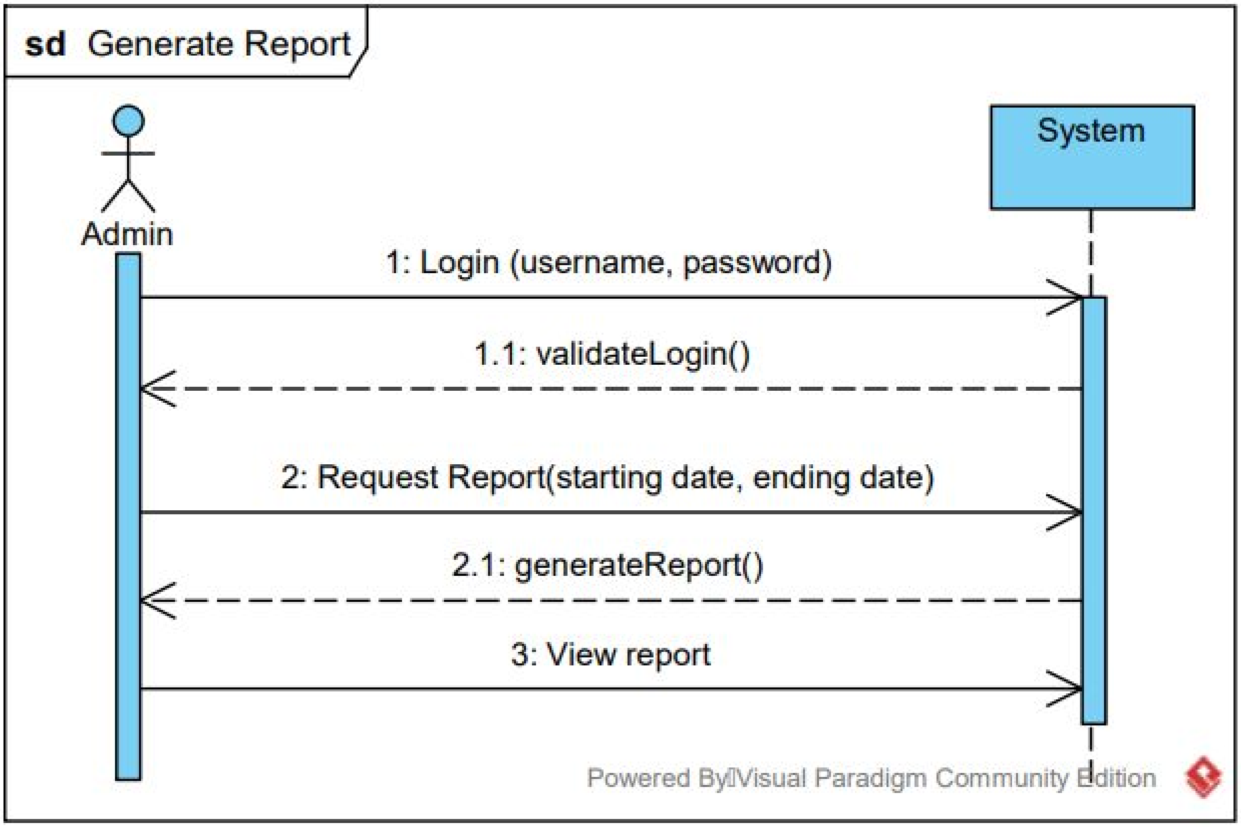
· 1.The admin reenters their credentials correctly, and are able to log on

3a. The date ending date occurred before the starting date

· 1.System prints an error, and Admin is prompted to enter date range

3b. The admin clicks cancel report

- 1.System clears page and goes back to main settings page


sd Generate Report
Admin
System
1: Login (username, password)
1.1: validateLogin()
2: Request Report(starting date, ending date)
2.1: generateReport()
3: View report
Powered By Visual Paradigm Community Edition

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 software has been created, with a number of posts created by users.

Success Guarantee (Postconditions): Rides that have been 100% confirmed, or expired past the date listed are removed from the database.

Main Success Scenario (or Basic Flow):

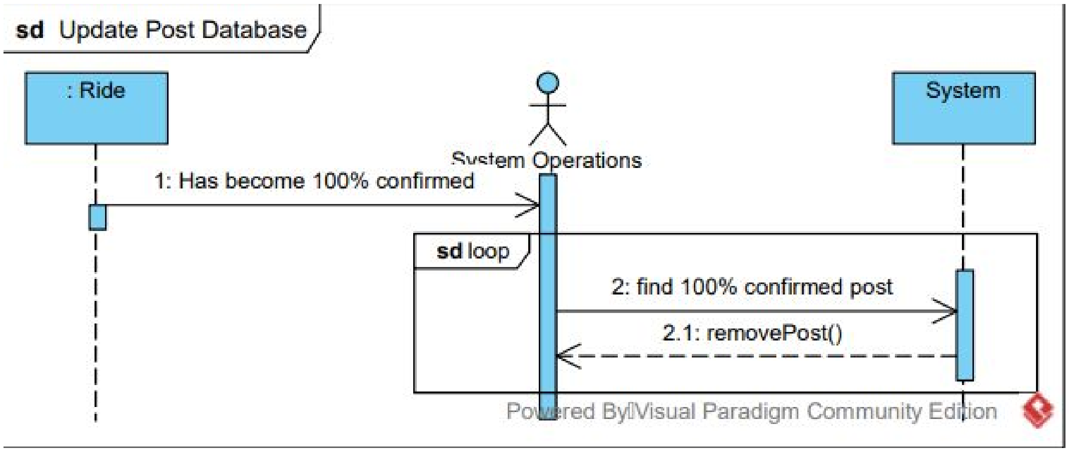
1. A ride has become 100% confirmed by all the users involved in a singular ride.
2. The system scans for posts that are 100% confirmed.
3. The system removes the data of the post from the database

Extensions (or Alternative Flows):

1a. The current date has past the ride date of a post.

· 1.The system scans for all posts that are behind the current date

2.The system removes the data of the post from the database


sd Update Post Database
: Ride
System Operations
System
1: Has become 100% confirmed
sd loop
2: find 100% confirmed post
2.1: removePost()
Powered By Visual Paradigm Community Edition

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): A user's profile has been removed or suspended under a certain circumstance.

Main Success Scenario (or Basic Flow):

1. A month has passed from a user's listed graduation month and year

2. The system scans the user profiles for accounts whose listed graduation dates are past the current date

3. All accounts that fit this criteria are removed, and the user database is updated.

Extensions (or Alternative Flows):

1a. A user has been reported for their driving.

· 1. The system receives the report and automatically suspends the account.

· 2.The user then has the option to appeal the suspension to be reinstated.

1b. A user has used inappropriate language in some capacity on the software

· 1. The system automatically removes accounts who have used severe language, threats, or derogatory language towards another person or group of people

1c. A user has selected to delete their account.

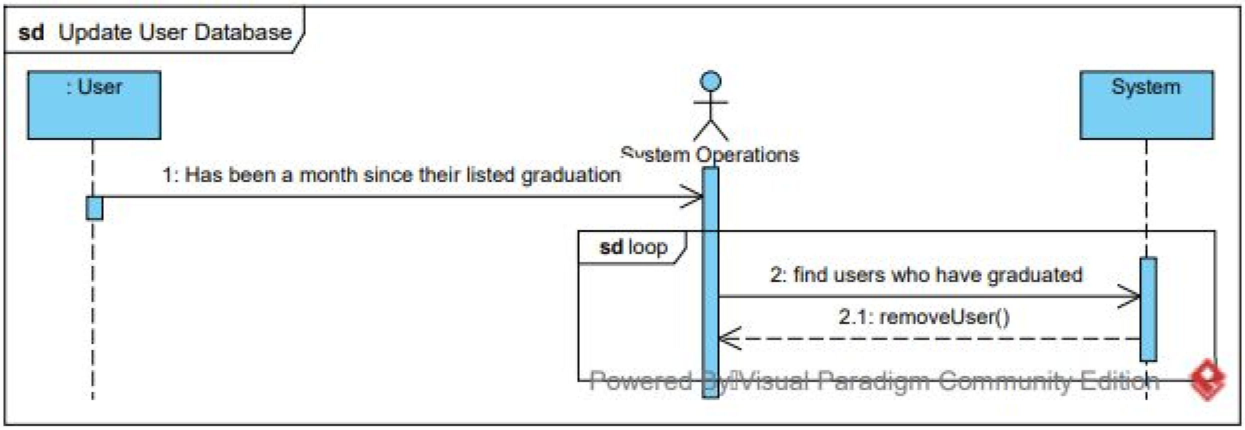
- 1. The system process the request, and prompts the user for a confirmation of deletion

- 2. The user confirms their request

- 3. The system removes the profile and its data from the database

1cb. The user cancels their confirmation to delete an account

- 1. The systems removes the request, and does not remove the profile


sd Update User Database
: User
System Operations
System
1: Has been a month since their listed graduation
sd loop
2: find users who have graduated
2.1: removeUser()
Powered By Visual Paradigm Community Edition

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 and the Driver has created a driver post.

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

Main Success Scenario (or Basic Flow):

1. The rider requires transportation from Point A to Point B.
2. The rider opens the application and clicks on "Driver Posts."
3. The rider clicks on the search icon
4. The application prompts the rider for "point A, point B, and date"
5. The application sorts the posts by information given
6. The rider scrolls through the sorted Driver Posts to find a suitable ride
7. The rider selects a ride.

Extensions (or Alternative Flows):

*a. At any time, System fails:

1. Close the application.

4a. User has not filled all prompted fields:

1. System sorts based on given fields.

4b. User has left all prompted fields empty:

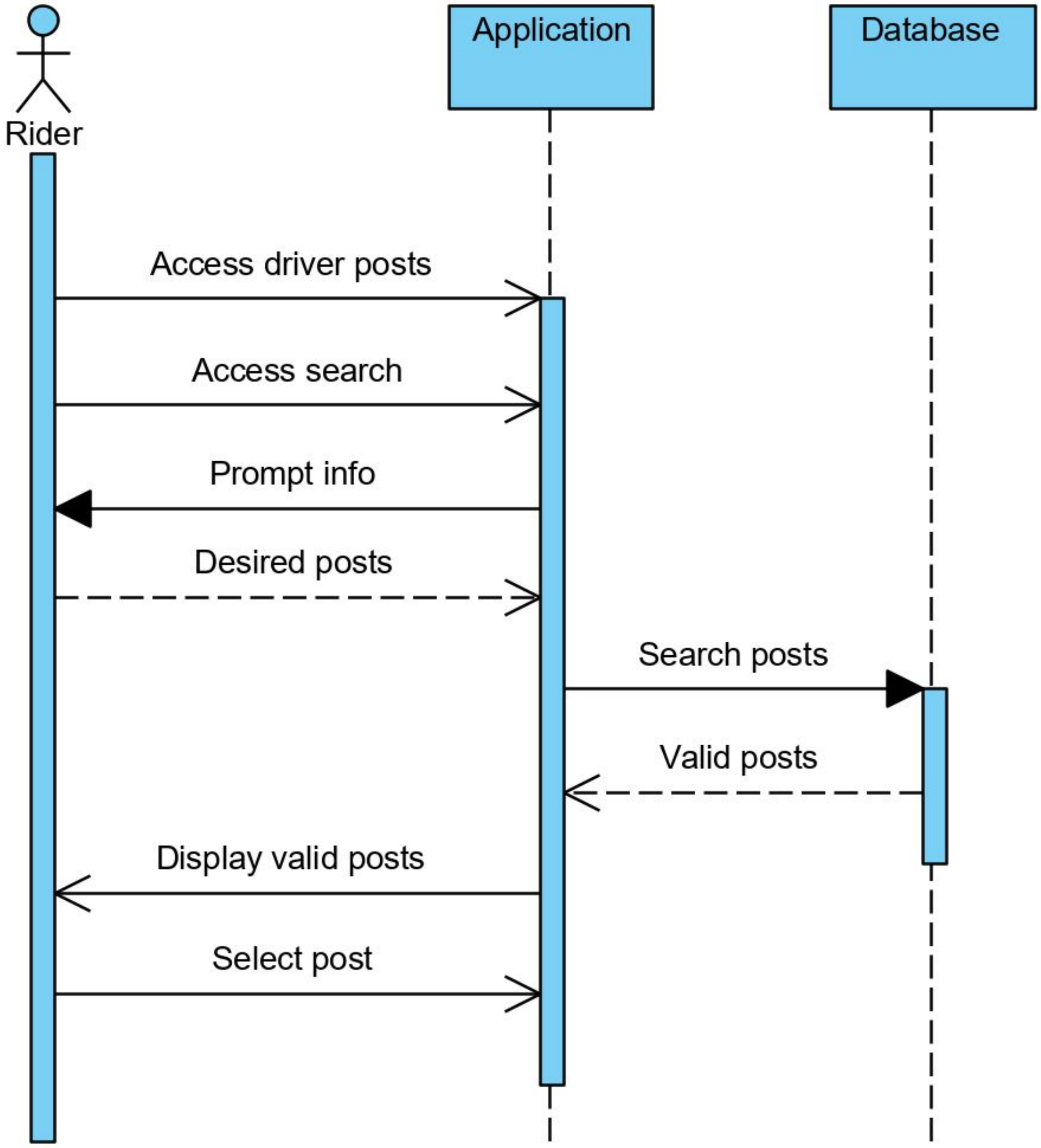
1. Application grays out and disables search button.

5a. No matching posts exist:

1. Display error message to screen
2. Prompt for another search

7a. User does not wish to select a driver:

1. User clicks the "X" button to go back to the main feed.


Rider
Application
Database
Access driver posts
Access search
Prompt info
Desired posts
Search posts
Valid posts
Display valid posts
Select post

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 and the Riders have created Rider Posts.

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

Main Success Scenario (or Basic Flow):

1. The driver is going to be driving from point A to point B.
2. The driver opens the application and clicks on "Rider Posts."
3. The driver clicks on the search icon
4. The application prompts the driver for "point A, point B, and date"
5. The application sorts the posts by information given
6. The driver scrolls through the sorted Rider Posts to find suitable riders
7. The driver selects a rider.

Extensions (or Alternative Flows):

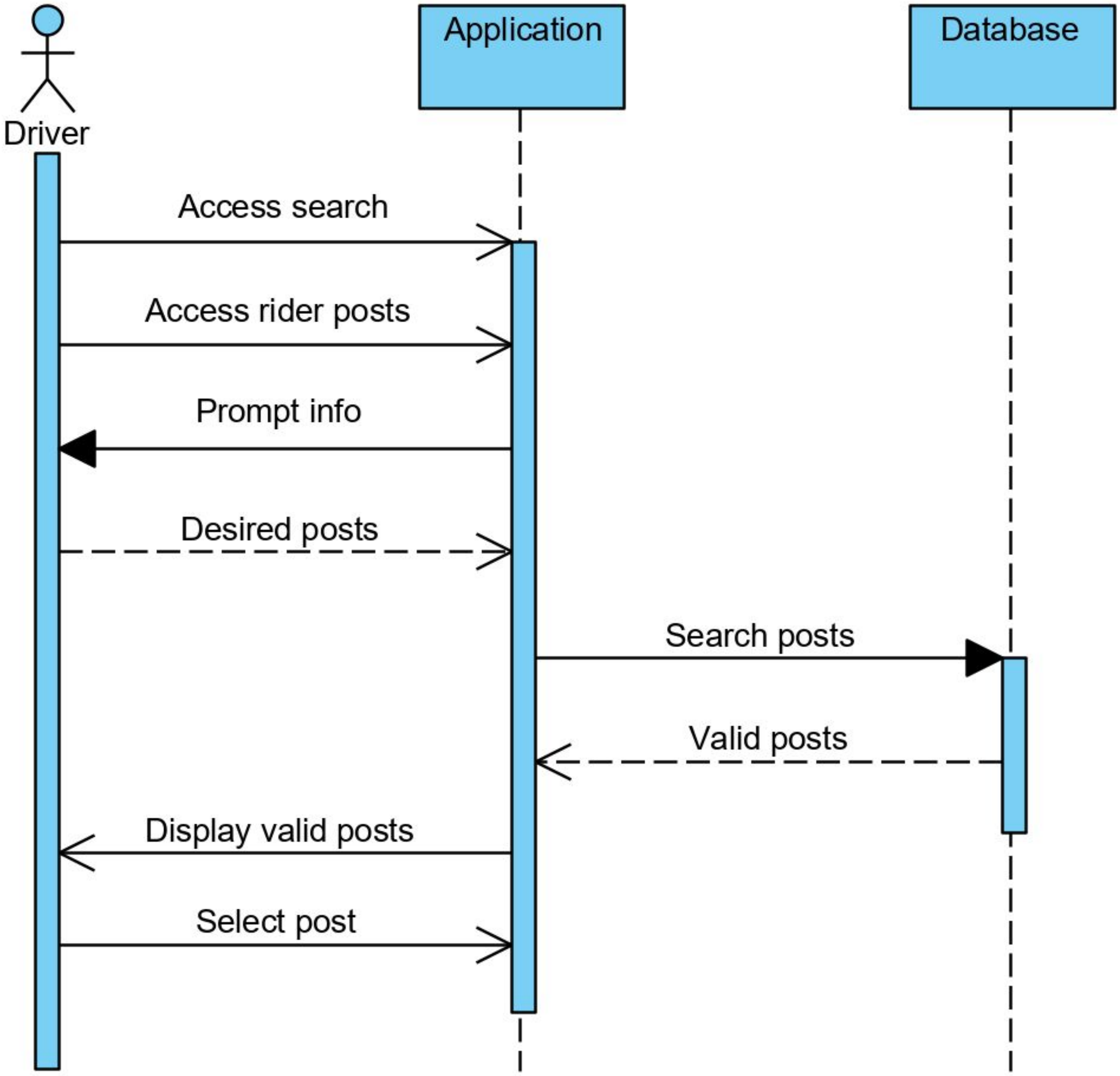
*a. At any time, System fails:
   1. Close the application.

4a. User has not filled all prompted fields:
   1. System sorts based on given fields.

4b. User has left all prompted fields empty:
   1. Application grays out and disables search button.

5a. No matching posts exist:
   1. Display error message to screen
   2. Prompt for another search

7a. User does not wish to select a rider:
   1. User clicks the "X" button to go back to the main feed.


Driver
Application
Database
Access search
Access rider posts
Prompt info
Desired posts
Search posts
Valid posts
Display valid posts
Select post

Use Case: Search database for post - Joshua Huertas

Scope: Baylor Rideshare finding a post

Level: Subfunction

Primary Actor: Application

Stakeholders & Interests:

- User: Wants to find specific posts easily.

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

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. User wants to search for a post with specific attributes
2. User clicks on the search icon
3. Application prompts for "point A, point B, and date"
4. Application accesses database
5. Application creates a set to hold the subset
6. Application populates set with posts that have the desired attributes
7. Application sorts the set based on time
8. Application displays the resulting set to the screen

Extensions (or Alternative Flows):

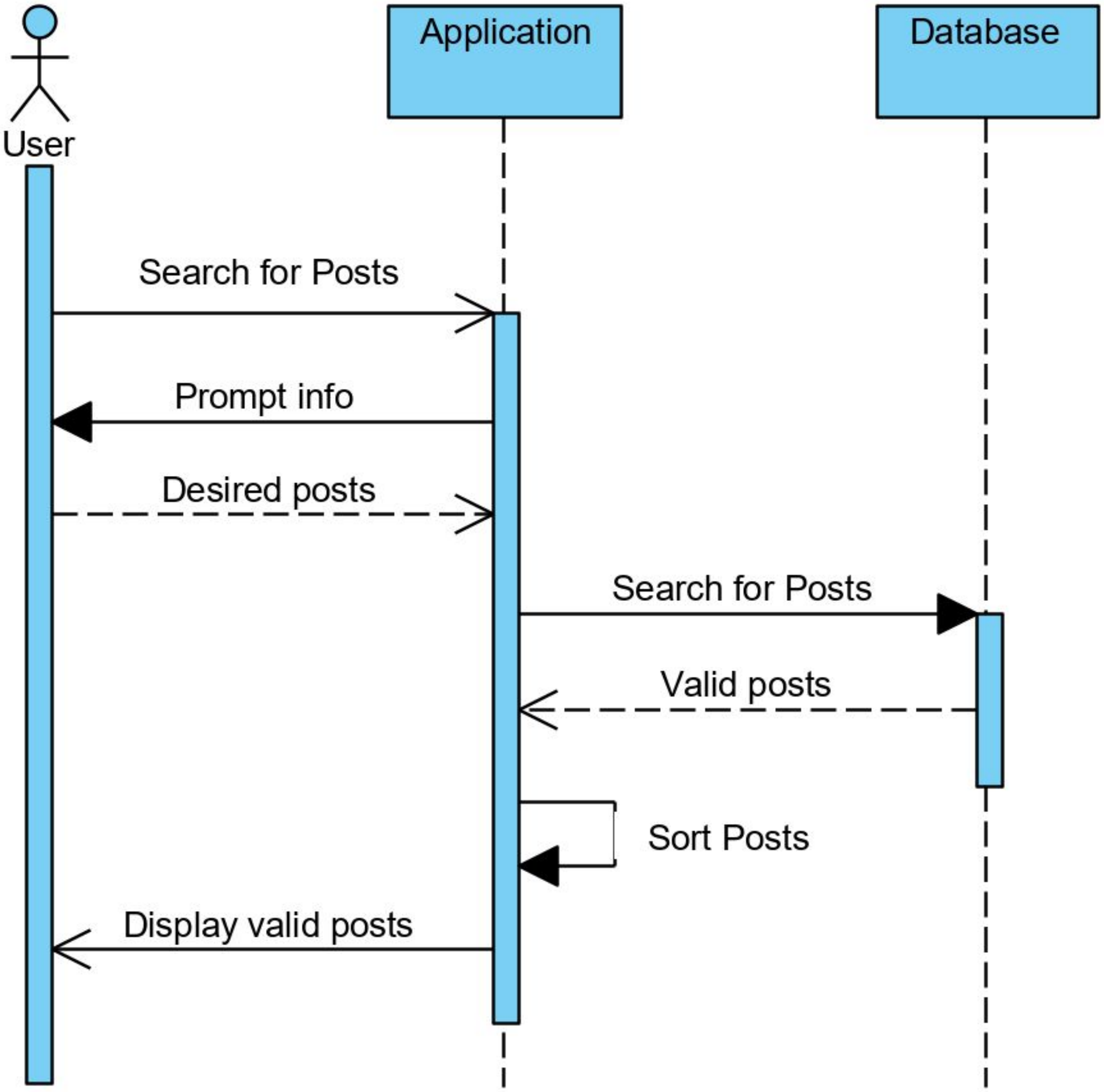
*a. At any time, System fails:
   1. Close the application.

3a. User has not filled all prompted fields:
   1. System sorts based on given fields.

3b. User has left all prompted fields empty:
   1. Application grays out and disables search button.

6a. No matching posts exist:
   1. Set remains empty

8a. The resulting set is empty:
   1. Display error message to the screen.
   2. Prompt for another search


User
Application
Database
Search for Posts
Prompt info
Desired posts
Search for Posts
Valid posts
Sort Posts
Display valid posts

Use case: Survey User - Joseph Perez

Scope: Survey

Level: Subfunction

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):

Admins will use data to improve overall quality of life experience. Criticism will be used to flesh out problems.

Main Success Scenario (or Basic Flow):

1. The user is asked if they would like to answer a survey.
2. User agrees and is prompted with questions.
3. User answers questions properly and submits the survey.
4. Answers delivered to database to be processed
5. GUI closes

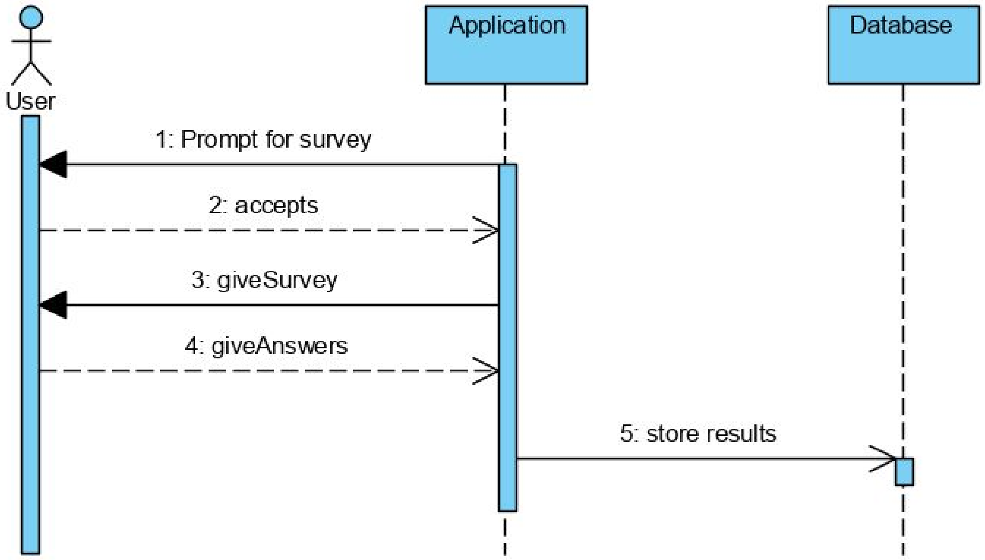
Extensions (or Alternative Flows):

a* The user declines the survey

1. The GUI closes

3a The user submits an incomplete survey

1. Let the user know that not all fields have been filled
2. Keep the information inputted and redisplay survey


User
Application
Database
1: Prompt for survey
2: accepts
3: giveSurvey
4: giveAnswers
5: store results

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 and driver exchange contact information.

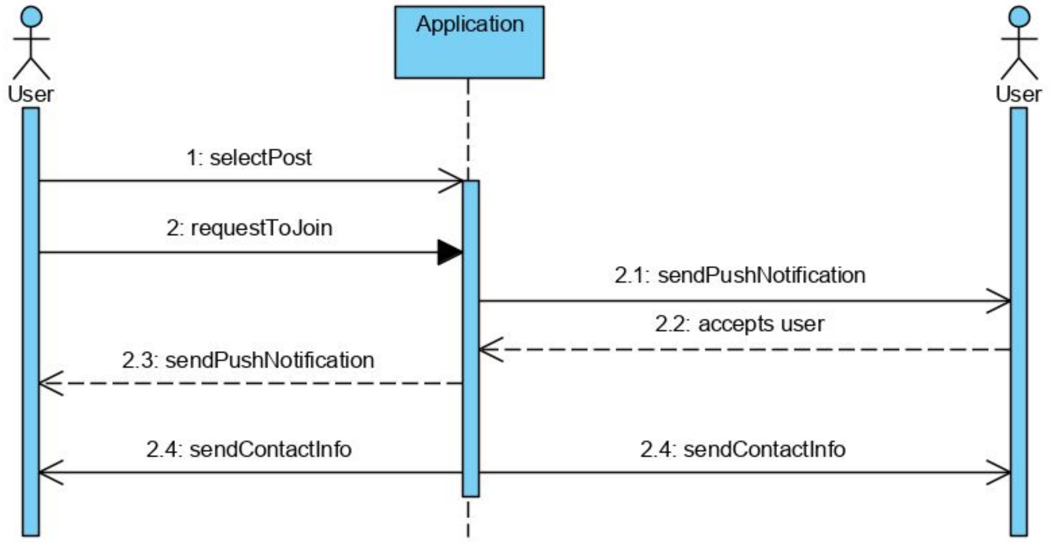
Main Success Scenario (or Basic Flow):

1. User selects a post.
2. User clicks request to join
3. System sends notification to poster
4. Poster views notification and accepts
5. User is notified their request was accepted
6. Users exchange information

Extensions (or Alternative Flows):

4a Poster views notification and declines

1. User is notified that their request post was declined


User
Application
User
1: selectPost
2: requestToJoin
2.1: sendPushNotification
2.2: accepts user
2.3: sendPushNotification
2.4: sendContactInfo
2.4: sendContactInfo

Use case: Remove Inappropriate posts - Joseph Perez

Scope: Inappropriate Behavior

Level: Admin goal

Primary Actor: Admin

Stakeholders & Interests:

- Users do not want to exhibit inappropriate behavior
- Admins want to make sure posts do not contain inappropriate content

Preconditions:

User has posted something that is potentially not in accordance with the application's posting guidelines.

Success Guarantee (Postconditions):

The post is removed and action is taken on the user at fault, the severity of action will be dictated by the admin.

Main Success Scenario (or Basic Flow):

1. Admin is notified about a questionable post.
2. Admin determines the post is not in compliance with the guidelines.
3. Admin takes action depending on severity.
4. User is notified that their post was removed due to inappropriate content.
5. User has the option to apply for reinstatement.

Extensions (or Alternative Flows):

1a. If a post is not flagged by the system

1. Users may report the post

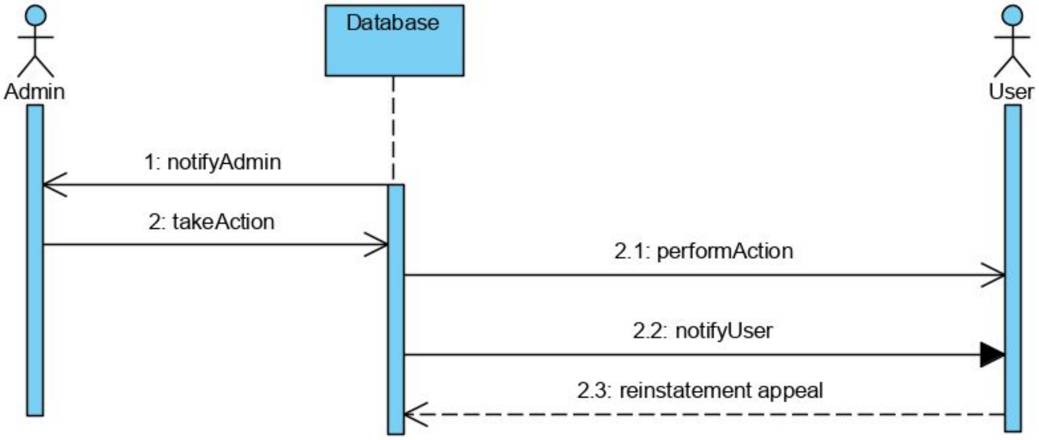
2. Admins will then be notified

2a. The post is in compliance with the guidelines

1. The flagged post is ignored

5a. If user's post is too severe

1. User is denied reinstatement


Admin
Database
User
1: notifyAdmin
2: takeAction
2.1: performAction
2.2: notifyUser
2.3: reinstatement appeal

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): User account is removed from database

Main Success Scenario (or Basic Flow):

1. User is logged in
2. User navigates to account tab
3. User selects “Delete Account”
4. System requests additional confirmation
5. User enters password
6. System removes user account from database
7. User is logged out, returned to login screen
8. All posts linked to user are removed

Extensions (or Alternative Flows):

4.a: User doesn’t select “Yes”

1. System aborts operation, user is returned to account tab

5.a: User enters incorrect password

1. System aborts operation, user is returned to account tab

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 accurate record of posts, must notify users affected by removal

Preconditions: User post exists in database

Success Guarantee (Postconditions): User post is removed from database, users are notified of post removal

Main Success Scenario (or Basic Flow):

1. User is logged in
2. User navigates to own posts
3. User selects “Delete Post”
4. System requests confirmation
5. User confirms post removal
6. System removes post from database
7. System notifies users affected by removal

Extensions (or Alternative Flows):

4.a: User doesn’t confirm

1. System aborts operation, returns user to post feed

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
- System: Wants accurate record of suspended user

Preconditions: User is not suspended

Success Guarantee (Postconditions): User has had permissions temporarily revoked

Main Success Scenario (or Basic Flow):

1. Admin is logged in
2. Admin searches for user to suspend
3. Admin reports reason for suspension
4. Admin confirms user suspension
5. System alerts user
6. System marks user as "Suspended"

Extensions (or Alternative Flows):

2.a: User cannot be found

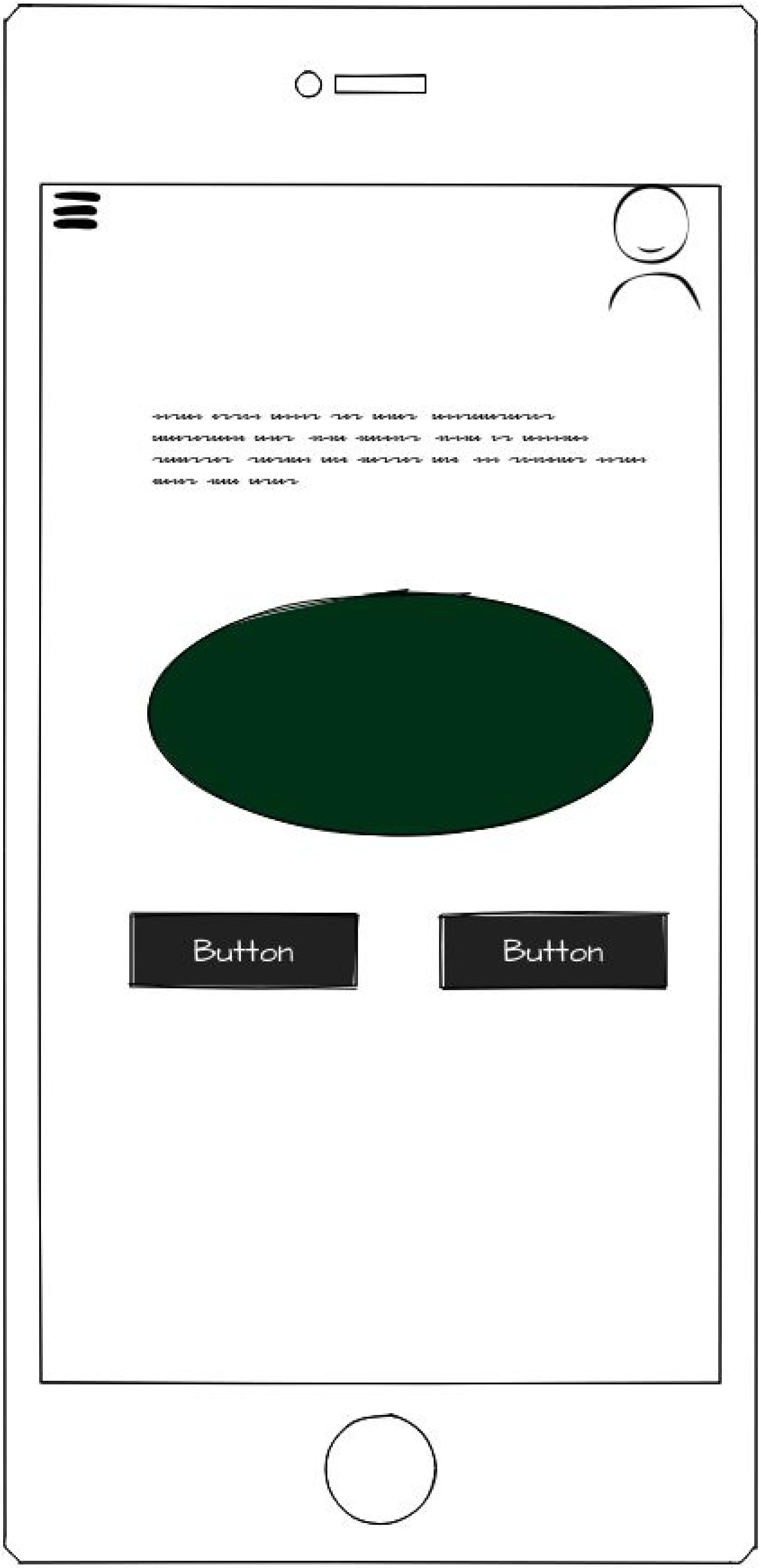
1. System alerts admin

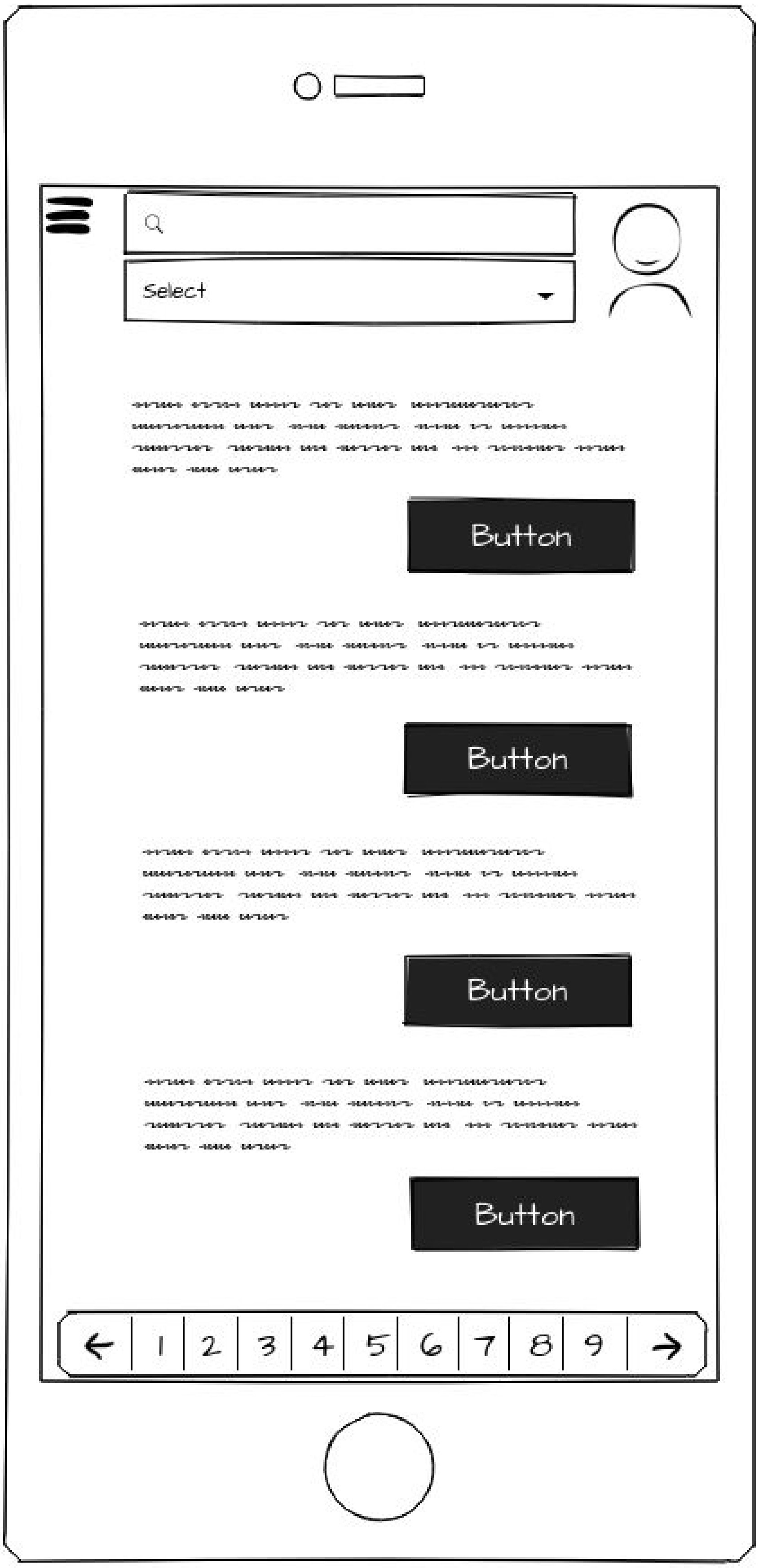
3.a: Reason left empty

1. System prompts admin to provide reason for suspension

4.a: Admin does not confirm

1. System aborts operation, admin returns to user search

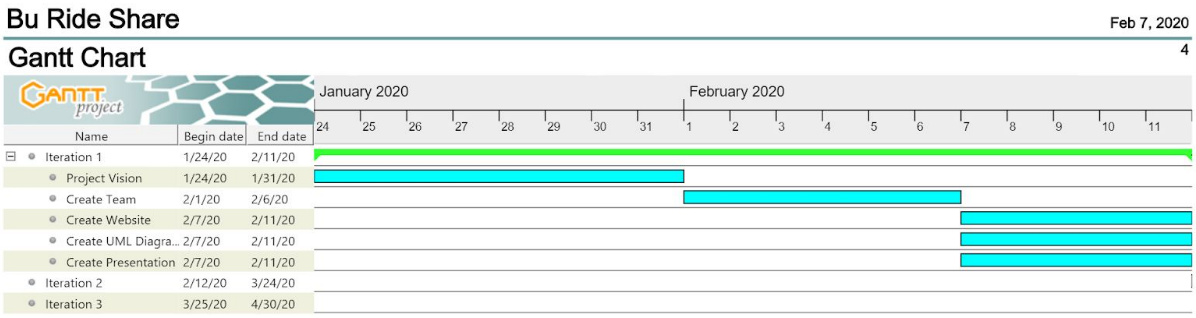

Button
Button


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

https://squirmyforcerny.github.io/CSI3471-Ride-Share/ - website

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

Gantt



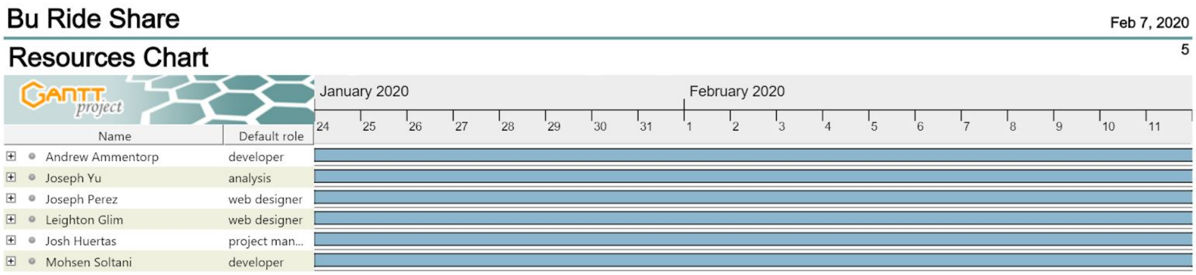
Bu Ride Share

Feb 7, 2020

Gantt Chart

4

Name	Begin date	End date
⊟ Iteration 1	1/24/20	2/11/20
Project Vision	1/24/20	1/31/20
Create Team	2/1/20	2/6/20
Create Website	2/7/20	2/11/20
Create UML Diagra...	2/7/20	2/11/20
Create Presentation	2/7/20	2/11/20
Iteration 2	2/12/20	3/24/20
Iteration 3	3/25/20	4/30/20



Bu Ride Share

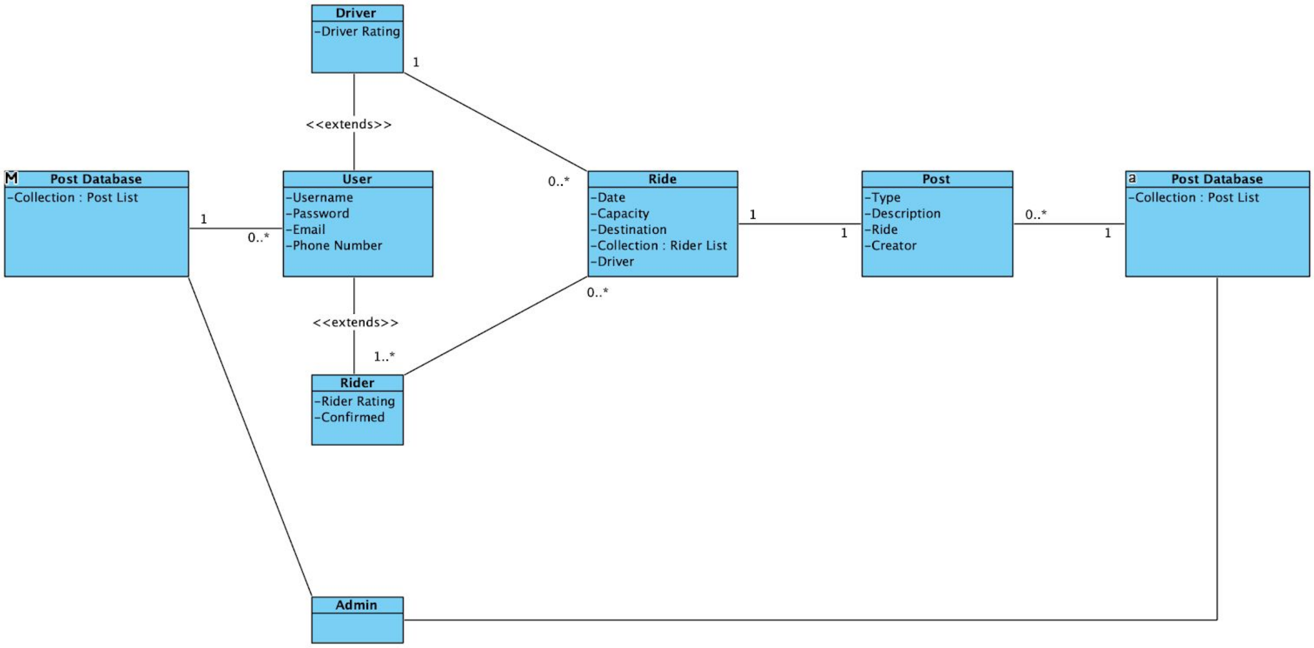
Feb 7, 2020

Resources Chart

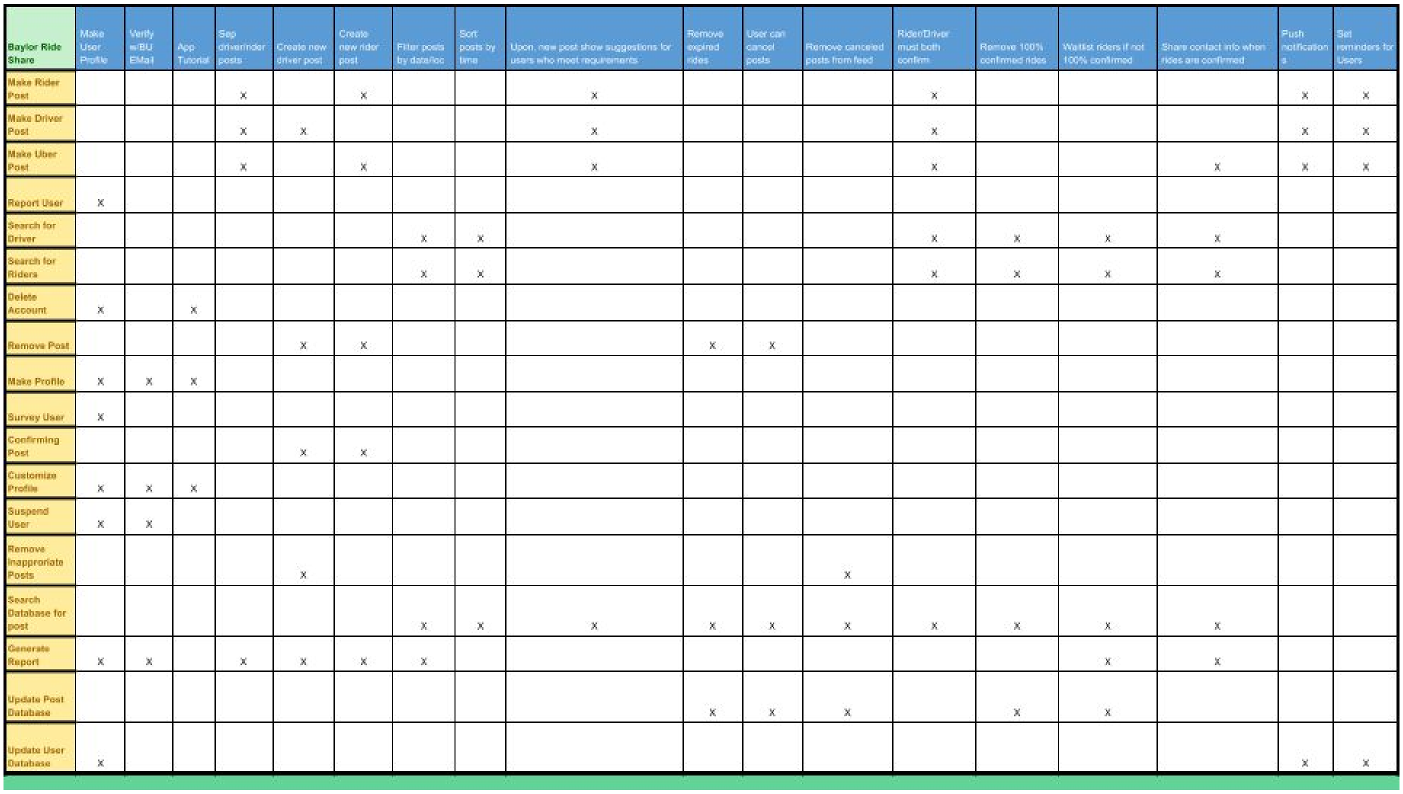
5

Name	Default role
⊞ Andrew Ammentorp	developer
⊞ Joseph Yu	analysis
⊞ Joseph Perez	web designer
⊞ Leighton Glim	web designer
⊞ Josh Huertas	project man...
⊞ Mohsen Soltani	developer

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 data/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