

# Senior Capstone Project

## Moove

Winter 2018

*Harsh Desai*

### Project Documentations:

#### 1. Project vision

##### 1.1. Backgrounds

- We realized that with the growth of ride sharing services like Uber and Lyft, these companies rely upon drivers who wish to work part time or full time. We wanted to make a service that provides rides to people for a cheaper rate than others, and with drivers who just want to go somewhere as well.

##### 1.2. Socio-economic Impact, Business Objectives, and Gap Analysis

- Socio-economic Impact
  - With ride sharing applications like Uber and Lyft blooming and growing everyday we feel like there is something missing. Something that other rideshare companies don't cater to. We want to cater to a more casual driver. We want to be able to have someone who is going to a destination anyways, pick up someone else who just so happens to want to go to the same place. We aren't necessarily only looking for drivers who want to drive for a living, but an average person who just wants to make a little extra money on the way to the place they wanted to go anyways. With this concept, the person looking for a ride can get one at a much cheaper rate than calling an Uber, Lyft, or cab; while also giving the driver some extra money for themselves on the way to their own destination, without inconveniencing any party.
- Business Objectives
  - Creating a stable product
  - Growing a user base/Marketing

- Creating a cheaper/fairer payment model for users
- Customer service
- Profitability
- User retention
- Ease of Access

- Gap Analysis

Objectives	Current State	Future State	Gap Description	Factors	Remedial Actions
<b>Ensure application works as intended</b>	Missing Simple elements, small bugs	Application complete	Not quite done yet	time	Spend an extra weekend fully fixing everything
Have 30 requests populated	Only 14 requests populated	40 requests populated	16 requests short	More DB space Not enough data	Populate more data

### 1.3. Security and ethical concerns

- Security
  - Securing payment and user data is our top priority
  - Securing users from outside sources (other users, accidents, theft, etc.) is a huge potential problem
- Ethical Concerns
  - How do we evaluate background checks?
  - How do we decide on the safest age to host a ride?
  - How do we validate drivers licenses?
  - How do we ensure our users safety?
  - How do we compensate for damage/accidents/theft?
  - What is our procedure for reviewing cases?
  - Who is held liable on a case by case basis?

### 1.4. Glossary of Key Terms

- Host - Term to describe a user who is hosting a ride, the driver of the car.
- Rider - Term used to describe a user who is getting a ride from someone, the passenger.
- User - Broad term that is only used when referring to both hosts and riders.

## 2. Project Execution and Planning

### 2.1. Team Information

- Adam Vida - Senior at OU studying IT. Experience with HTML, CSS, Java, C++ and Android application development.
- Harsh Desai - Senior at OU studying Computer Science. Experience in MySQL, Java, C++, C, HTML, CSS and PHP
- Sagun Sedai - Senior at OU studying Computer Science. Have experience in Java, PHP, HTML, CSS, SQL, JavaScript, C, C++, and C# languages.
- Wesley Shall - Senior at OU studying Computer Science. Experienced in Android/ Web Development
- Nathaniel Auxier - Senior at OU studying Computer Science. Have experience in Java, PHP, HTML, C#, SQL, JavaScript, C, C++, Python languages, Web Development, and API integration
- Apurva Tolia - Senior at OU studying IT. Experienced in designing the UI and writing complex algorithm for the application.

### 2.2. Tools and Technology

- Tools
  - XAMPP
  - Android Studio
  - PHPStorm
  - BootStrap
  - Apache Server
  - Ubuntu 16.10 OS
  - Digital Ocean server
  - Stripe Payment API
  - Google Distance Matrix API
  - Google Waypoint API
- Technology
  - CSS
  - HTML
  - PHP
  - JavaScript
  - JQuery
  - MySQL (PHPMyadmin)

### 2.3. Project Plan

- The big picture scope of the project is meant to include a web application that allows for users to ask for rides and to drive other riders. This application also includes methods for users to pay their drivers. Along with

the web application, an android mobile application will also be created to provide the same experience as the web application but on the go. (\*Update: Due to time constraints, the mobile application has been scraped, only the web application will be created).

#### 2.4. Best standards and Practices

- Since a web application for Uber and Lyft already exist, it seems the best models for the website should be somewhat inspired by those designs so that we can create a better ride sharing application. In addition, the standard that we set is to be competitive in the market with companies such as Uber and Lyft by having cheaper rates, and secure services so that the user can enjoy the most comfortable experience.

### 3. System Requirement Analysis

#### 3.1. Function Requirements

ID	Priority Level (1"higher" to 5 "lower")	Requirement
REQ 1	1	The user must have an internet connection to access our application.
REQ 2	1	The user must register on the website or mobile app in order to host or join a ride.
REQ 3	1	The user must login to the website or mobile app using a username and password.
REQ 4	1	The web application and mobile application must always be synced up and be functioning almost identically.
REQ 5	1	The host should be over 21 and must have 2 years of clean driving record.
REQ 6	1	The host must be able to host the ride successfully.
REQ 7	3	The host must be able to change the destination mid-ride.
REQ 8	1	The host must be able to manually end a ride prematurely or once the destination has been reached.
REQ 9	1	The host must be able to submit a request to charge a user more money if damage is done to the car or the driving route is changed to be further than originally agreed.

<b>REQ 10</b>	1	The rider must be able to request a ride.
<b>REQ 11</b>	1	The rider must receive a confirmation message when a booking has been requested.
<b>REQ 12</b>	3	The host must be able to view the booking request made
<b>REQ 13</b>	3	The rider will be able to see the status of the driver.
<b>REQ 14</b>	1	The rider must be able to provide payment using a credit card, debit card or paypal before using the service.
<b>REQ 15</b>	3	The user must be able to securely save payment card information on the app for future purchases.
<b>REQ 16</b>	2	The rider must be able to appeal an extra charge if they feel they were unfairly charged.
<b>REQ 17</b>	5	The rider is allowed to take limited amount of luggage depending on the host car.
<b>REQ 18</b>	3	The host must receive a confirmation email after creating an account for the first time.
<b>REQ 19</b>	1	Hosts must be able to set parameters such as certain times, dates, and certain miles away from pickup points that they are able to accept for giving rides to people.
<b>REQ 20</b>	1	Account must be deleted upon user request.
<b>REQ 21</b>	5	The host must provide a time frame the rider has to enter the car before the driver can drive off and continue his routes.
<b>REQ 22</b>	3	The rider must receive two confirmation messages when the host has arrived to the pickup location and drop off location.
<b>REQ 23</b>	2	The user should be able to see a history/log of his or hers past requests and drop-offs that have been made. In order to easily make the same requests repeatedly.
<b>REQ 24</b>	1	The rider must be able to see the total charges of the entire trip that they are requesting before they agree to pay.

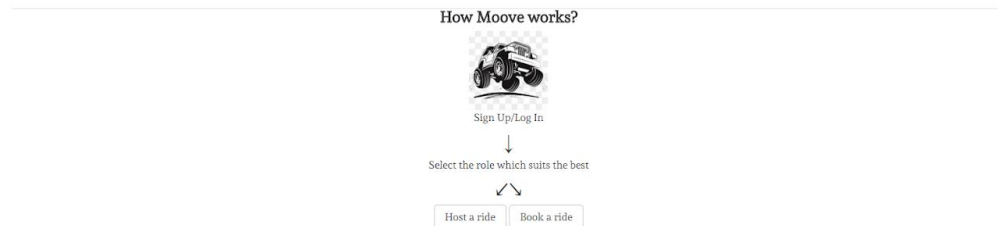
### 3.2. Non-functional Requirements


ID	Priority Level (1"higher" to 5 "lower")	Requirement
REQ 1	1	The rider should be able to access the ride service in many locations across the United States.
REQ 2	1	The host should be able to view the location of the customer in order to provide the service.
REQ 3	1	The rider should be able to provide payment to the host after they have reached their destination.
REQ 3	2	The rider should be able to receive the confirmation message of their booking within 10 minutes.
REQ 4	2	The rider should be allowed to cancel their booking if they don't want to use the service but a fee may apply.
REQ 5	3	The rider should be able to rate a host after a ride has ended.
REQ 6	3	The rider should be able to leave a comment on the host after a ride has ended.
REQ 7	3	The host should be able to rate a user after a ride has ended.
REQ 8	4	The user should be able to connect their account with paypal, or another third party payment service.
REQ 9	5	The user should get email notification about the deals and offers if the user has turned on the subscription.
REQ 10	4	The rider should be able to the real time map where they can see their driver location.
REQ 11	3	The rider should receive different rates if they are above 65 + or over.
REQ 12	1	The host should have a background check completed prior to performing the service.
REQ 13	1	If the rider is 18 years or younger requesting for a ride, then a present adult who is 18 years or older should accompany them in the ride.
REQ 14	1	The user should be prohibited to carry firearms while in the vehicle.

<b>REQ 15</b>	2	The host should briefly instruct the passenger the terms and conditions of using Moove.
<b>REQ 16</b>	2	The rider should be limited to the amount of luggage that they have in the vehicle.
<b>REQ 17</b>	2	User should be able to see the security policy in order to give them confidence that they can trust the storage of their personal payment information on the Moove's database (example: 128 bit encryption, with masked account numbers, etc).
<b>REQ 18</b>	1	User should have option of going through the process to become a driver. Provided the user meets the driver requirements provided by Moove.
<b>REQ 19</b>	1	The host should strictly be enforced before signing the terms and conditions that absolutely no drugs or alcohol is allowed while operating any vehicle.

### 3.3. On-Screen Appearance of landing and other pages requirements.

Moove is a ride-sharing application where the driver can host the ride and the passenger(s) can join the host and share the ride.









Adam Vida

No file chosen

[About Us](#)
[Contact Us](#)
[FAQs](#)[Host a ride](#)
[Book a ride](#)
[Site Map](#)

Copyright © 2018 Moove

Hosted Rides	
2018-04-13 12:00 AM Seats Available - 1 Vehicle : HONDA	Hilton Auburn Hills Suites, Featherstone Road, Auburn Hills, MI, USA to Hyatt Place Detroit/Auburn Hills, North Opdyke Road, Auburn Hills, MI, USA <div> <input type="button" value="View Map"/>  </div> <div> <input type="button" value="Accept"/> <input type="button" value="Delete"/> </div>
2018-03-27 08:30 AM Seats Available - 4 Vehicle : Honda Civic	Starbucks, Orchard Lake Road, Farmington Hills, MI, USA to Star Deli, Twelve Mile Road, Southfield, MI, USA <div> <input type="button" value="View Map"/>  </div> <div> <input type="button" value="Accept"/> <input type="button" value="Delete"/> </div>
2018-03-27 03:00 PM Seats Available - 3 Vehicle : Honda Civic	Starbucks, Orchard Lake Road, Farmington Hills, MI, USA to Starbucks, Orchard Lake Rd, West Bloomfield Township, MI, USA <div> <input type="button" value="View Map"/>  </div> <div> <input type="button" value="Accept"/> <input type="button" value="Delete"/> </div>
2018-03-27 04:00 PM Seats Available - 6 Vehicle : Dodge Durango	Kroger, Orchard Lake Road, Orchard Lake Village, MI, USA to Costco Wholesale, Commerce Crossing, Commerce Charter Township, MI, USA <div> <input type="button" value="View Map"/>  </div>



Moove

Innovate the World

⚙️

🏠

Logout

Map

Satellite

Start: Hilton Auburn Hills Suites, Featherstone Road, Auburn Hills, MI, USA

End: Hyatt Place Detroit/Auburn Hills, North Opdyke Road, Auburn Hills, MI, USA

Submit

Route Segment: 1

2300 Featherstone Rd, Auburn Hills, MI 48326, USA to 4301 Orchard Lake Rd, West Bloomfield Township, MI 48323, USA

10.0 mi

Route Segment: 2

4301 Orchard Lake Rd, West Bloomfield Township, MI 48323, USA to 1967 Pine Ridge Ln, Bloomfield Hills, MI 48302, USA

About Us

Contact Us

FAQs

f

🐦

▶️

📷

Copyright © 2018 Moove

Log In

Sign Up

Site Map

Moove Service

First Ride

✕

✉️ Email

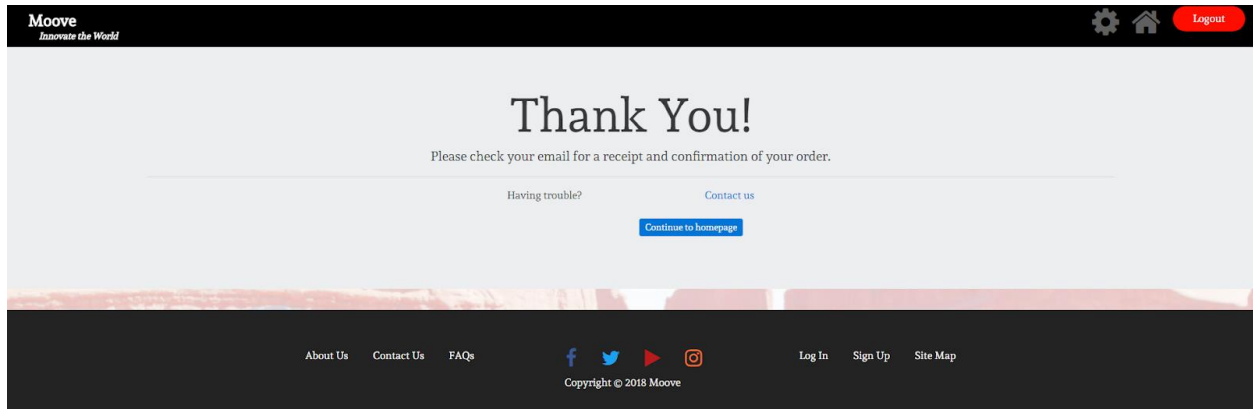
💳 Card number

📅 MM / YY

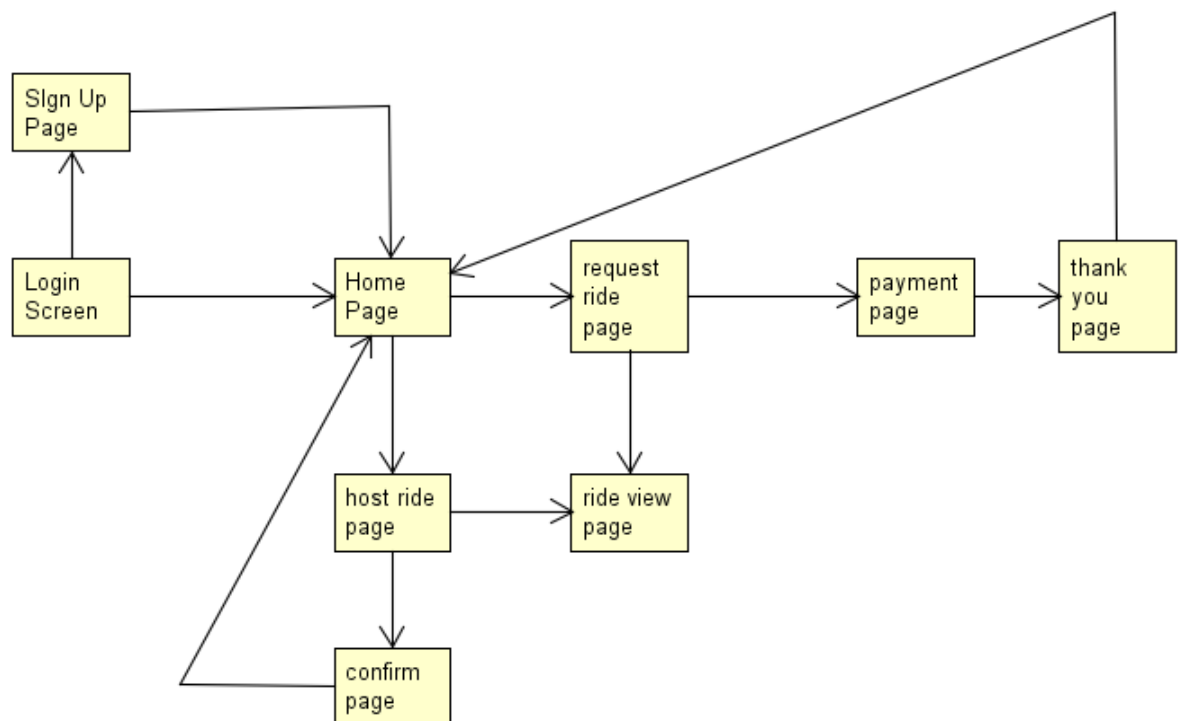
🔒 CVC

☐ Remember me

Pay \$5.00



### 3.4. Wireframe designs



## 4. Functional Requirements Specification

### 4.1. Stakeholders

- Uber and Lyft - These companies will become competitors and may end up having to adjust their policies in response to Moove or vice versa.
- Car companies (Ford, Chrysler, etc) - These companies could set up sponsorships deals or other such things. A new company revolving around cars and travel could always impact car companies.
- US Government - A very serious stakeholder, there are many liabilities that are implied with a company like Moove and along with Uber and Lyft, these

companies may spark new legislation in direct response to the rise in ride sharing companies.

#### 4.2. Actors and Goals

- Actors
  - Rider
  - Host
  - Database
  - Application
- Goals
  - Rider - To successfully request and receive a ride from a host.
  - Host - To successfully receive ride requests and get riders to their destination.
  - Database - To hold data of all users in a secure and efficient manner.
  - Application - To facilitate the means to provide the ride sharing service to all users.

#### 4.3. User stories, scenarios and Use Cases

- User Stories

US 1	As a rider, I can select a destination I want to go to and book a ride
US 2	As a host, I can view all available booking requests, and fulfill ones I would like
US 3	As a host, I will receive payment upon completing a ride request
US 4	As a host, I can click a button to broadcast that I am hosting rides
US 5	As a host, I can end hosting a ride once a ride completes
US 6	As a host, I can change the original destination in the middle of the ride
US 7	As a user, I can add a profile picture and some basic information about me
US 8	As a user, I can apply to become a host by submitting my drivers license and performing a background check

US 9	As a rider, I can set up payment methods to be used to pay for the rides I receive.
---------	---

- Critical Scenarios
- *Scenario 1: User decides to register to use the application*
  - John Smith opens the web or mobile application
  - He observes that there are two options, which are Login and Sign Up
  - John picks the “Sign Up” button.
  - Then, he has to setup his First Name, Last Name, email address, password, and Date of Birth
  - After that he clicks “Sign Up” button.
- *Scenario 2: User decides to login to use the application*
  - John Smith opens the web or mobile application
  - He observes that there are two options, which are Login and Sign Up
  - Then, he has to input his email address and password that he registered
  - After that he clicks “Login” button..
- *Scenario 3: User decides to logout from the application*
  - John Smith is using the ride-share application and wants to quit.
  - He presses the “Logout” button on the mobile or web application.
- *Scenario 4: Rider decides to view the available rides*
  - While John Smith is using the application, he decides that he wants to see what rides are available.
  - He inputs his location and the destination that he wants to go to.
  - Then, he presses the “View Available Rides” button.
  - He observes a list of drivers who are available to pick him up from his current location.
- *Scenario 5: Rider decides to book a ride*
  - After John Smith viewed the available rides, he observes the name of the driver and the rate of the ride.
  - John decides to select that option for preparation of booking and is shown to another page where there are terms and conditions of the ride.
  - Then, he accepts it, and selects the “Book now” button.
  - John receives a confirmation text stating that he booked the ride.
  - The driver also receives information by text that a passenger needs to be picked up.
- *Scenario 6: Host decides to host a ride*
  - After John Smith logs in, he decides to host a ride himself.
  - John navigates to the Hosting page and selects “Host Now”
  - John has not registered himself as a Driver yet so he is prompted to provide additional details about himself

- After providing a valid driver license number, insurance policy number, and vehicle VIN, John selects Submit.
  - John is then prompted that he is now hosting and other users can view his driver information and book a ride
- *Scenario 7: Host decides to accept ride request*
  - After John Smith receives a ride request, he is prompted with a notification that a passenger needs to be picked up, detailing the passenger's details and their location.
  - John selects "Accept" and is presented a map with a GPS route to the passenger's location.
  - The passenger also receives a notification that the driver has accepted their request is on route.
- *Scenario 8: Host fulfils ride request*
  - After John Smith picks the passenger up, he proceeds to the destination
  - After reaching the destination and the passenger exits, he selects "Complete Ride"
  - He is presented with the total amount received for the run that is then deposited into his account
  - He selects "Continue" and is prompted that he is available again.
- *Scenario 9: Host ends hosting a ride*
  - John Smith decides he is finished hosting
  - He selects "End Hosting" and is notified that he is not longer hosting.
- *Scenario 10: User withdraws money from account*
  - John Smith decides it is pay day and wishes to withdraw what he made during the week
  - After logging in, he selects "Account Balance" and is presented with the amount that is currently in his account
  - He selects "Withdraw" and is presented with options on how he would like to receive payment, whether be a direct deposit to a bank account or by check in mail.
  - John selects "Check" and confirms his address and that he wishes to receive the money by check.

- Use Cases

Use Case 1	Authentication
Related Requirements	REQ 1, REQ 2, REQ 3, REQ 4
Initiating Actor	User
Actor's Goal	To grant access to the connected client to the website/app
Participating Actors	User, Login process, internal DB
Preconditions	<ul style="list-style-type: none"> <li>User has to be registered on the site</li> </ul>
Postconditions	<ul style="list-style-type: none"> <li>Connected client is given access to the site</li> </ul>
Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>1. User enters in username and password credentials</li> <li>2. Login process retrieves the values of the email address and password fields</li> <li>3. Login process performs comparisons with internal database for matching email address and password record</li> <li>4. Once matching record is found, login process grants</li> </ol>
Flow of Events for Alternate Scenario	<p>3a. If no matching record is found, login process denies access and prompts "Invalid email address or password" to connected client</p>

Use Case 2	Account Creation
Related Requirements	REQ1,18
Initiating Actor	User
Actor's Goal	Successfully create an account
Participating Actors	User, internal DB
Preconditions	<ol style="list-style-type: none"> <li>1. User has a working internet connection and has opened the application</li> </ol>
Postconditions	<ol style="list-style-type: none"> <li>1. User has the ability to login</li> </ol>
Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>1. User clicks on create account button</li> <li>2. User enters account information such as username, password, email, etc.</li> <li>3. User receives confirmation email</li> <li>4. User clicks email link to activate account</li> </ol>
Flow of Events for Alternate Scenario	<ol style="list-style-type: none"> <li>1. User clicks on create account button</li> <li>2. User enters bad or already used account information</li> <li>3. User is denied a confirmation email and a message saying invalid email/username/etc.</li> </ol>

<b>Use Case 3</b>	<b>Login</b>
<b>Related Requirements</b>	REQ1, REQ2, REQ3, REQ4
<b>Initiating Actor</b>	User
<b>Actor's Goal</b>	Successfully log into the site
<b>Participating Actors</b>	User, Internal DB
<b>Preconditions</b>	User is registered
<b>Postconditions</b>	User is logged in
<b>Flow of Events for Main Success Scenario</b>	<ol style="list-style-type: none"> <li>1. Enter username and password then click Login</li> <li>2. User is logged into the site</li> </ol>
<b>Flow of Events for Alternate Scenario</b>	<ol style="list-style-type: none"> <li>1. Enter invalid username and password then click Login</li> <li>2. Invalid username or password message appears</li> </ol>



Use Case 4	Account Deletion
Related Requirements	REQ 2, REQ 20
Initiating Actor	User
Actor's Goal	Successfully remove and deactivate account
Participating Actors	User, internal DB, confirmation request
Preconditions	<ul style="list-style-type: none"> <li>User has to be registered on the site</li> <li>User has to be logged in</li> </ul>
Postconditions	<ul style="list-style-type: none"> <li>Connected client is given access to the site</li> </ul>
Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>Under settings, user must select delete account</li> <li>Prompted with a message "Are you sure you want to delete your account?" the user must select confirm</li> <li>Message will be prompted that states "Your account has been successfully deleted"</li> </ol>
Flow of Events for Alternate Scenario	<p>3a. Error Occurs while deletion is in progress</p> <p>3b. Displays message reading "An error has occurred while processing, please try again later."</p>

Use Case 5	Host booking a ride
Related Requirements	REQ1, 2, 3, 4, 5, 6, 7, 8, 12, 19
Initiating Actor	Host user
Actor's Goal	Successfully accept a ride request
Participating Actors	Host user, Rider user, application, internal DB
Preconditions	<ol style="list-style-type: none"> <li>Host is logged in and registered as a valid driving host.</li> <li>User ride requests are available to be reviewed.</li> <li>Host has set parameters for who they can pick up</li> </ol>

<b>Postconditions</b>	<ol style="list-style-type: none"> <li>1. A ride has been booked and user is awaiting their ride</li> </ol>
<b>Flow of Events for Main Success Scenario</b>	<ol style="list-style-type: none"> <li>1. Host clicks on available bookings for hosts button</li> <li>2. Based on host parameters, qualifying requests are displayed.</li> <li>3. Host chooses a request.</li> <li>4. Request rider is notified via email</li> <li>5. Request rider confirms details.</li> <li>6. Host clicks notify button during driving time to send notification to rider that their ride is on the way</li> <li>7. Once rider is dropped off, host hits end drive button.</li> <li>8. Payment is exchanged via application automatically</li> <li>9. drive ends and host is brought back to main screen</li> </ol>
<b>Flow of Events for Alternate Scenario</b>	<ol style="list-style-type: none"> <li>1. Host clicks on available bookings for hosts button</li> <li>2. Based on host parameters, qualifying requests are displayed.</li> <li>3. Host chooses a request.</li> <li>4. Request rider is notified via email</li> <li>5. Request rider confirms details.</li> <li>6. Host clicks notify button during driving time to send notification to rider that their ride is on the way</li> <li>7. Rider decides to change destination mid-ride</li> <li>8. Host hits change route button</li> <li>9. Host enters in new destination</li> <li>10. Alert is sent to rider, who then receives a new cost estimate and confirms destination change.</li> <li>11. Once rider is dropped off, host hits end drive button.</li> <li>12. Payment is exchanged via application automatically</li> <li>13. drive ends and host is brought back to main screen</li> </ol>

<b>Use Case 6                      Removing a hosted ride</b>	
<b>Related Requirements</b>	REQ1,2, 3, 4, 5, 6, 7, 8
<b>Initiating Actor</b>	Host
<b>Actor's Goal</b>	Successfully delete a ride that was being hosted
<b>Participating Actors</b>	Host, internal DB
<b>Preconditions</b>	<ol style="list-style-type: none"> <li>1. Host is logged in as host</li> <li>2. The host is broadcasting as a host for rides to users</li> </ol>
<b>Postconditions</b>	<ol style="list-style-type: none"> <li>1. Host has removed his broadcast for other users to view</li> </ol>
<b>Flow of Events for Main Success Scenario</b>	<ol style="list-style-type: none"> <li>1. Host navigates to options menu</li> <li>2. Host clicks end host button</li> <li>3. Host clicks a confirmation button to ensure they want to end broadcasting</li> </ol>

	4. The host is no longer broadcasting their services
Flow of Events for Alternate Scenario	<ol style="list-style-type: none"> <li>1. Host navigates to options menu</li> <li>2. Host clicks end host button</li> <li>3. A popup notification appears that says the host already has a ride booked and that the user who is booked will get a full refund for a ride</li> <li>4. A possible penalty will be added to the hosts account for canceling a booking</li> <li>5. Host clicks a confirmation button to ensure they want to end broadcasting</li> <li>6. The host is no longer broadcasting their services</li> </ol>

Use Case 7	Requesting a Ride
Related Requirements	REQ1, 2, 3, 4, 10, 11, 12, 13, 14
Initiating Actor	Rider
Actor's Goal	Successfully request a ride from a host
Participating Actors	Rider, Host, application, internal DB
Preconditions	<ol style="list-style-type: none"> <li>1. Rider is registered and logged into their account with a valid credit/debit card saved to the account</li> </ol>
Postconditions	<ol style="list-style-type: none"> <li>1. Rider has received a booking confirmation and is awaiting their ride</li> </ol>
Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>1. Rider clicks request ride button</li> <li>2. Rider enters destination</li> <li>3. Rider enters time and date for the ride</li> <li>4. Any potential hosts are found and a cost estimate is displayed for the user</li> <li>5. Rider confirms payment information</li> <li>6. The rider clicks a final confirm ride button</li> <li>7. The host confirms the ride by clicking an accept ride button</li> <li>8. An email confirmation of successful booking is sent to rider</li> </ol>
Flow of Events for Alternate Scenario	<ol style="list-style-type: none"> <li>1. Rider clicks request ride button</li> <li>2. Rider enters destination</li> <li>3. Rider enters time and date for the ride</li> <li>4. No hosts are found to give a ride for that destination</li> </ol>

	<ol style="list-style-type: none"> <li>The request remains broadcasted to hosts until it is accepted by one</li> <li>Once a host accepts, an email is sent to the rider saying that the ride has been accepted and booked</li> <li>A cost estimate is displayed for the rider</li> <li>Rider confirms payment information</li> <li>Rider clicks a final confirm ride button.</li> <li>A confirmation message is sent to the rider when the host is on its way</li> </ol>
--	--

Use Case 8	Canceling a Ride as Rider
Related Requirements	REQ 1, REQ 2, REQ 8, REQ 10, REQ 11, REQ 13, REQ 16
Initiating Actor	Rider
Actor's Goal	Cancelling a requested ride
Participating Actors	Login process, internal DB, Rider
Preconditions	<ul style="list-style-type: none"> <li>Rider has to be logged in</li> <li>Rider has to have an active requested ride with Host</li> </ul>
Postconditions	<ul style="list-style-type: none"> <li>Rider is granted with confirmation within the application that the ride has been cancelled</li> </ul>
Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>Rider clicks on rider options</li> <li>Rider selects cancel ride</li> <li>Rider confirms that they would like to cancel</li> <li>Message will be sent to the host and rider that the ride has been cancelled</li> <li>Rider will be prompted to the main screen</li> </ol>
Flow of Events for Alternate Scenario	<ol style="list-style-type: none"> <li>Rider clicks on rider options</li> <li>Rider selects cancel ride</li> <li>Rider confirms they would like to cancel</li> <li>Rider loses network connection</li> <li>Driver arrives to destination</li> <li>After 5 minute waiting period the Host can cancel the ride</li> </ol>

Use Case 9                      Browsing for a ride	
Related Requirements	REQ 1, REQ 2, REQ 3, REQ 4, REQ 10
Initiating Actor	Rider
Actor's Goal	Successfully search for the ride that they want to join
Participating Actors	Rider, internal DB
Preconditions	<ol style="list-style-type: none"> <li>1. Rider should have a working internet connection.</li> <li>2. Rider should open the application in the mobile app or website.</li> </ol>
Postconditions	<ol style="list-style-type: none"> <li>1. Rider has successfully searched for the ride that they wanted to join.</li> </ol>
Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>1. The rider clicks on the rides section in the app or website.</li> <li>2. Then, the rider inputs their destination that they want to go from their location.</li> <li>3. Rider receives a list of the rides that are available near their location.</li> <li>4. Rider has the option to select the hosted ride that they prefer.</li> </ol>
Flow of Events for Alternate Scenario	<ol style="list-style-type: none"> <li>1. The Rider clicks on the rides section in the app or website.</li> <li>2. Then, the rider inputs their destination that they want to go from their location.</li> <li>3. However, no rides are available so the user receives a message that "No rides are available near their location at this time".</li> <li>4. rider can search again to see if there are any available rides.</li> </ol>

Use Case 10                      Joining a ride	
Related Requirements	REQ 1, REQ 2, REQ 3, REQ 4, REQ 10, REQ 13
Initiating Actor	Rider

<b>Actor's Goal</b>	Successfully join a ride when they had requested
<b>Participating Actors</b>	User, internal DB
<b>Preconditions</b>	<ol style="list-style-type: none"> <li>1. Rider should have a working internet connection.</li> <li>2. Rider should open the application in the mobile app or website.</li> </ol>
<b>Postconditions</b>	<ol style="list-style-type: none"> <li>1. Rider has successfully joined the ride that they requested.</li> </ol>
<b>Flow of Events for Main Success Scenario</b>	<ol style="list-style-type: none"> <li>1. The rider on the mobile app or website clicks on the book ride button after completing all of the payment information.</li> <li>2. The host would receive a notification on their phone that the user has requested for a ride.</li> <li>3. Then, it is updated to the website and app that particular host is now booked.</li> <li>4. The rider receives confirmation message that they has joined the ride.</li> </ol>
<b>Flow of Events for Alternate Scenario</b>	<ol style="list-style-type: none"> <li>1. The rider on the mobile app or website clicks on the book ride button after completing all of the payment information.</li> <li>2. The host would receive a notification on their phone that the user has requested for a ride.</li> <li>3. However, for some reason the host decides to cancel their booking.</li> <li>4. Then, there would be an update on the website and mobile app that the host is available to give a ride.</li> <li>5. The rider receives a notification that the booking has been canceled.</li> </ol>

<b>Use Case 11</b>	<b>Adding Payment Method</b>
<b>Related Requirements</b>	REQ 1, REQ 2, REQ 3, REQ 4, REQ 14, REQ 15
<b>Initiating Actor</b>	User
<b>Actor's Goal</b>	Successfully add the payment method.
<b>Participating Actors</b>	User, internal DB
<b>Preconditions</b>	<ol style="list-style-type: none"> <li>1. User should have a working internet connection.</li> <li>2. User should open the application in the mobile app or website</li> </ol>
<b>Postconditions</b>	<ol style="list-style-type: none"> <li>1. User has successfully added the payment method</li> </ol>

Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>1. User navigates to the payment options under the settings tab</li> <li>2. User add the payment method.</li> <li>3. User receives confirmation message on screen that payment method is added.</li> </ol>
Flow of Events for Alternate Scenario	<ol style="list-style-type: none"> <li>1. User navigates to the payment options under the settings tab</li> <li>2. User enters bad or incorrect payment information</li> <li>3. User receives a message saying invalid card holder name/card number/incorrect cvv number..</li> </ol>

Use Case 12	Removing Payment Method
Related Requirements	REQ 1, REQ 2, REQ 3, REQ 4, REQ 14, REQ 15
Initiating Actor	User
Actor's Goal	Successfully removed the payment method.
Participating Actors	User, internal DB
Preconditions	<ol style="list-style-type: none"> <li>1. User should have a working internet connection.</li> <li>2. User should open the application in the mobile app or website</li> </ol>
Postconditions	<ol style="list-style-type: none"> <li>1. User has successfully removed the payment method</li> </ol>
Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>1. User navigates to payment options under the settings tab</li> <li>2. User must enter their password.</li> <li>3. User clicks to remove the payment method.</li> <li>4. User receives confirmation message on screen that payment method is successfully removed</li> </ol>

Flow of Events for Alternate Scenario	<ol style="list-style-type: none"> <li>1. User navigates to payment options under the settings tab</li> <li>2. User must enter their password</li> <li>3. User enters bad or incorrect password to remove the payment method</li> <li>4. User receives a message saying invalid password in order to remove your payment method please enter your password again.</li> </ol>
---------------------------------------	--

Use Case 13	Adding Profile Detail
Related Requirements	REQ 1, 3
Initiating Actor	User
Actor's Goal	Successfully add profile detail
Participating Actors	User, Internal DB
Preconditions	<ol style="list-style-type: none"> <li>1. User has already created an account</li> <li>2. User has internet connection and either access to the app or the website</li> </ol>
Postconditions	<ol style="list-style-type: none"> <li>1. User has successfully added profile detail</li> </ol>
Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>1. User accesses profile settings</li> <li>2. User changes profile settings</li> <li>3. User receives in app notification that the changes have been saved</li> <li>4. User can visibly see the changes after saving them</li> </ol>
Flow of Events for Alternate Scenario	<ol style="list-style-type: none"> <li>1. User accesses profile settings</li> <li>2. User changes profile settings</li> <li>3. User receives in app notification that the changes have not been saved due to internet connection</li> <li>4. User cannot visibly see the changes after attempting to save them</li> </ol>

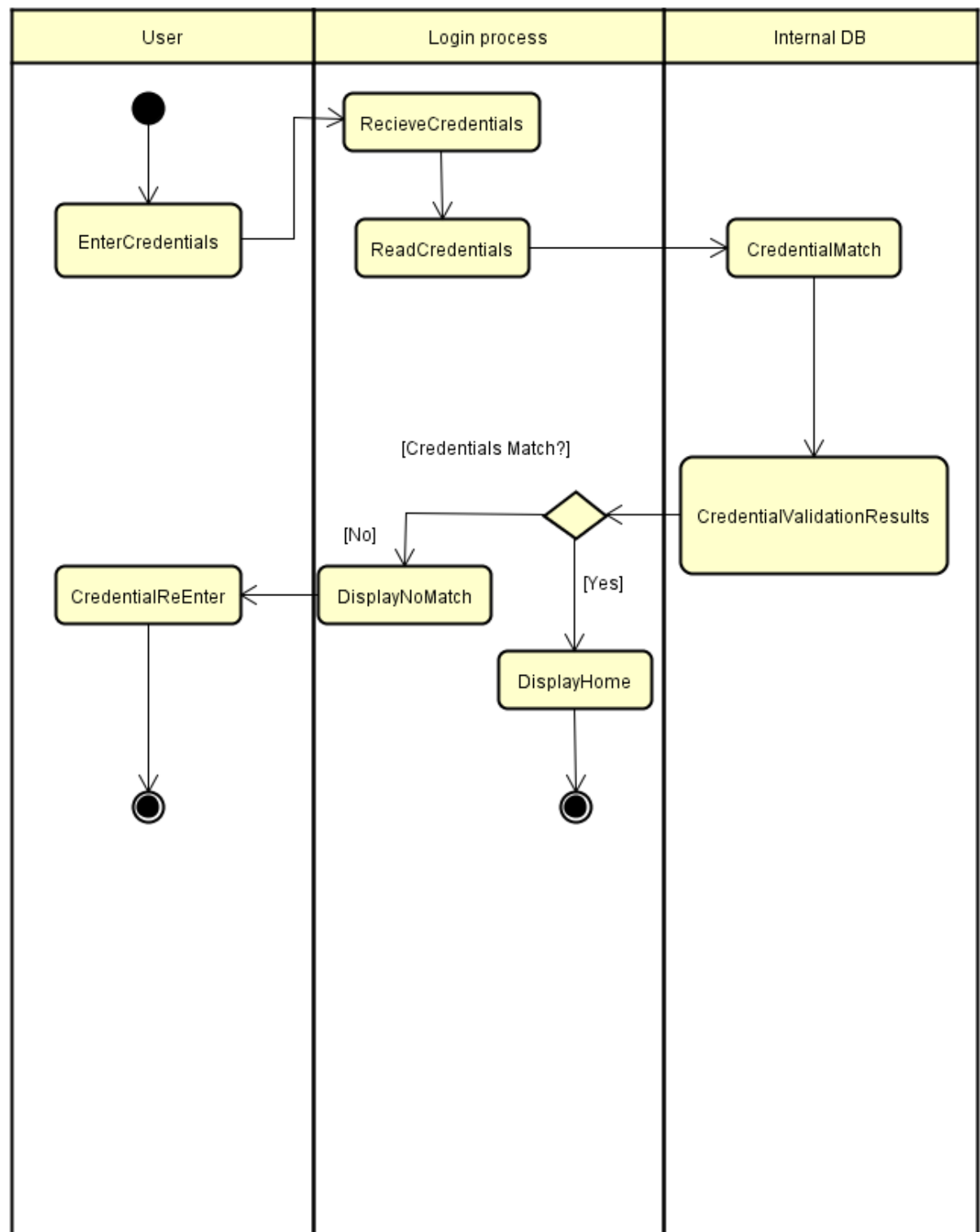


Use Case 14                      Removing Profile Detail	
Related Requirements	REQ 1, 3
Initiating Actor	User
Actor's Goal	Successfully remove profile detail
Participating Actors	User, Internal DB
Preconditions	<ol style="list-style-type: none"> <li>1. User has already created an account</li> <li>2. User has internet connection and either access to the app or the website</li> </ol>
Postconditions	<ol style="list-style-type: none"> <li>1. User has successfully removed profile detail</li> </ol>
Flow of Events for Main Success Scenario	<ol style="list-style-type: none"> <li>1. User accesses profile settings</li> <li>2. User changes profile settings</li> <li>3. User receives in app notification that the changes have been saved</li> <li>4. User can visibly see the changes after saving them</li> </ol>
Flow of Events for Alternate Scenario	<ol style="list-style-type: none"> <li>1. User accesses profile settings</li> <li>2. User changes profile settings</li> <li>3. User receives in app notification that the changes have not been saved due to internet connection</li> <li>4. User cannot visibly see the changes after attempting to save them</li> </ol>

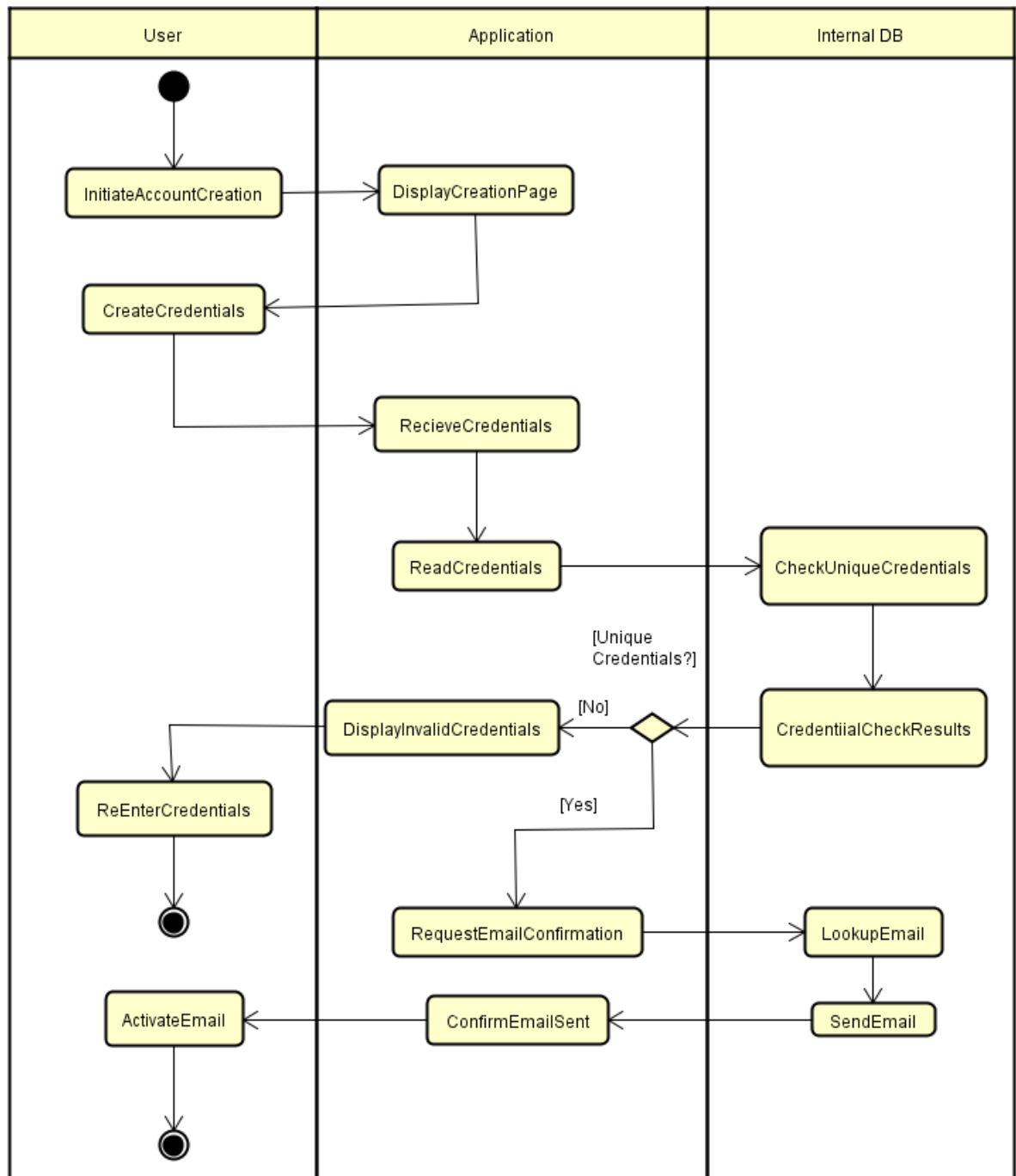
<b>Use Case 15</b>	<b>Receiving an email</b>
<b>Related Requirements</b>	REQ 1, REQ 2,REQ 18
<b>Initiating Actor</b>	User
<b>Actor's Goal</b>	Successfully receive an email.
<b>Participating Actors</b>	User, internal DB
<b>Preconditions</b>	<ol style="list-style-type: none"> <li>1. User had to be registered to the site in order to receive an email.</li> </ol>
<b>Postconditions</b>	<ol style="list-style-type: none"> <li>1. User has successfully received an email.</li> </ol>
<b>Flow of Events for Main Success Scenario</b>	<ol style="list-style-type: none"> <li>1. User register for a new account.</li> <li>2. User gets an confirmation email in order to activate their account.</li> <li>3. User clicks on activate my account now and the account is successfully activated.</li> </ol>
<b>Flow of Events for Alternate Scenario</b>	<ol style="list-style-type: none"> <li>1. User register for a new account.</li> <li>2. User enters bad or incorrect password or incorrect email.</li> <li>3. User receives a message saying invalid password /email Id in order to add an account please enter your valid email Id/password again.</li> </ol>

#### 4.4. System Sequence / Activity Diagrams

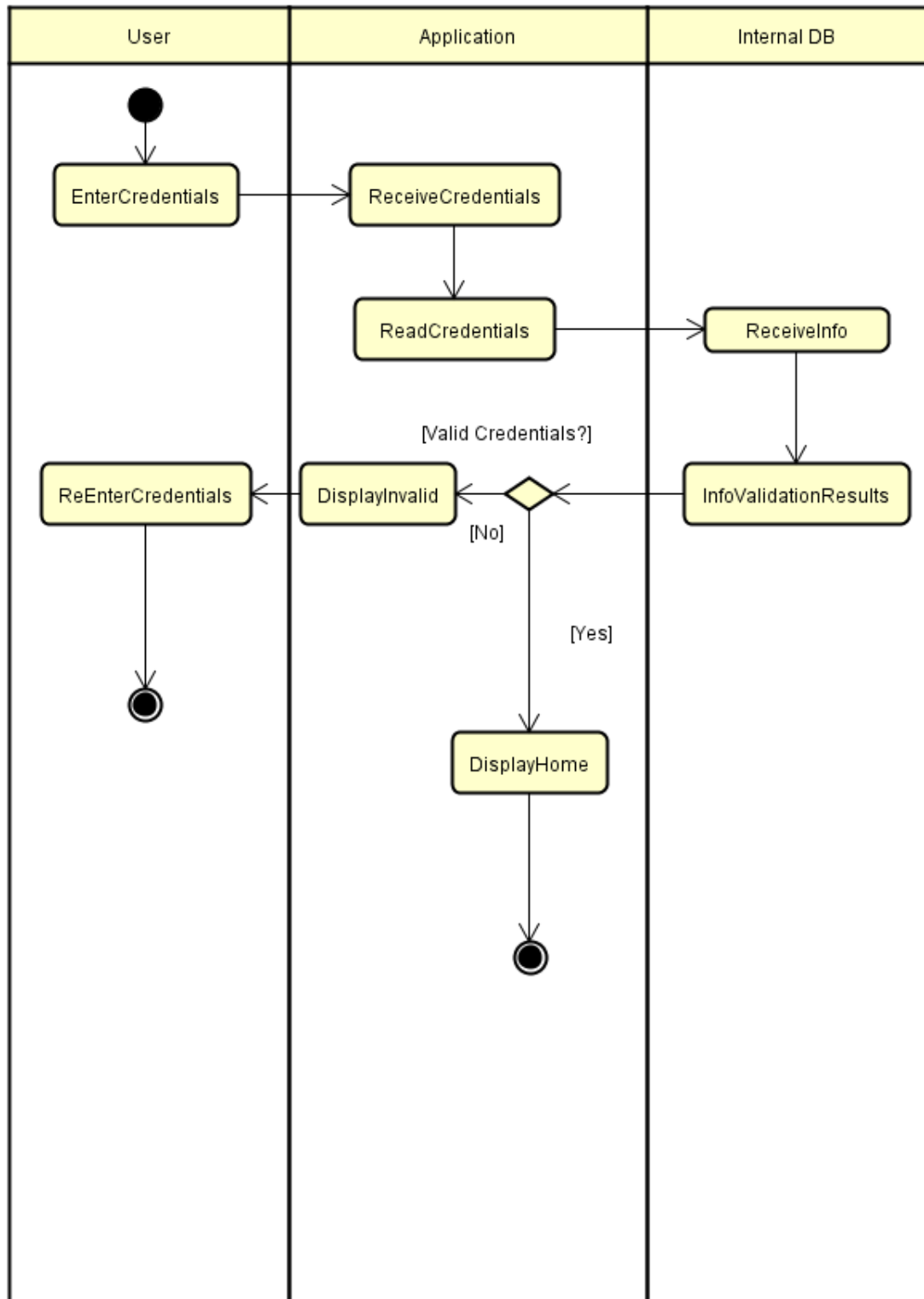
actAD1 - Authentication



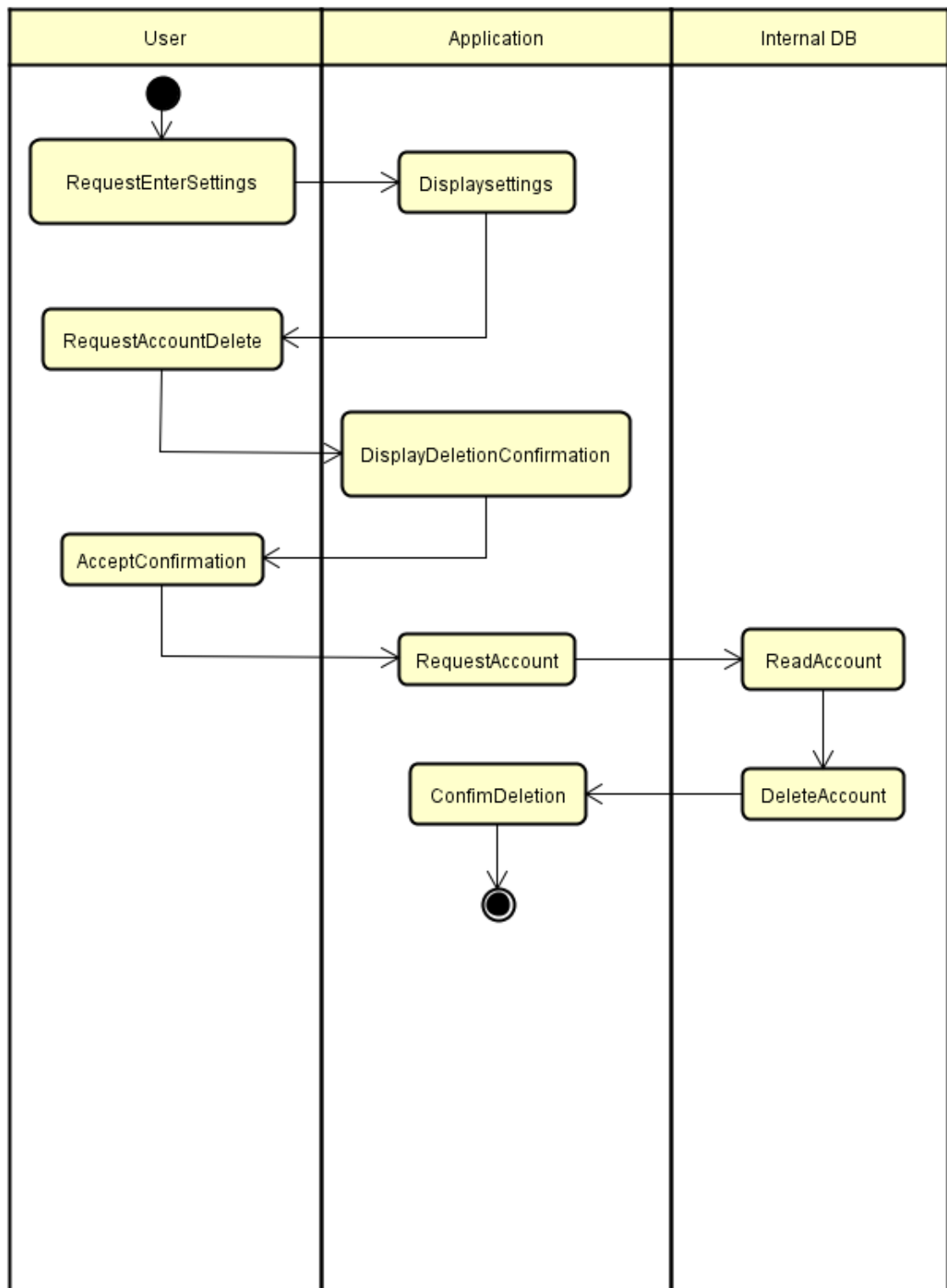
actAC2 - Account Creation

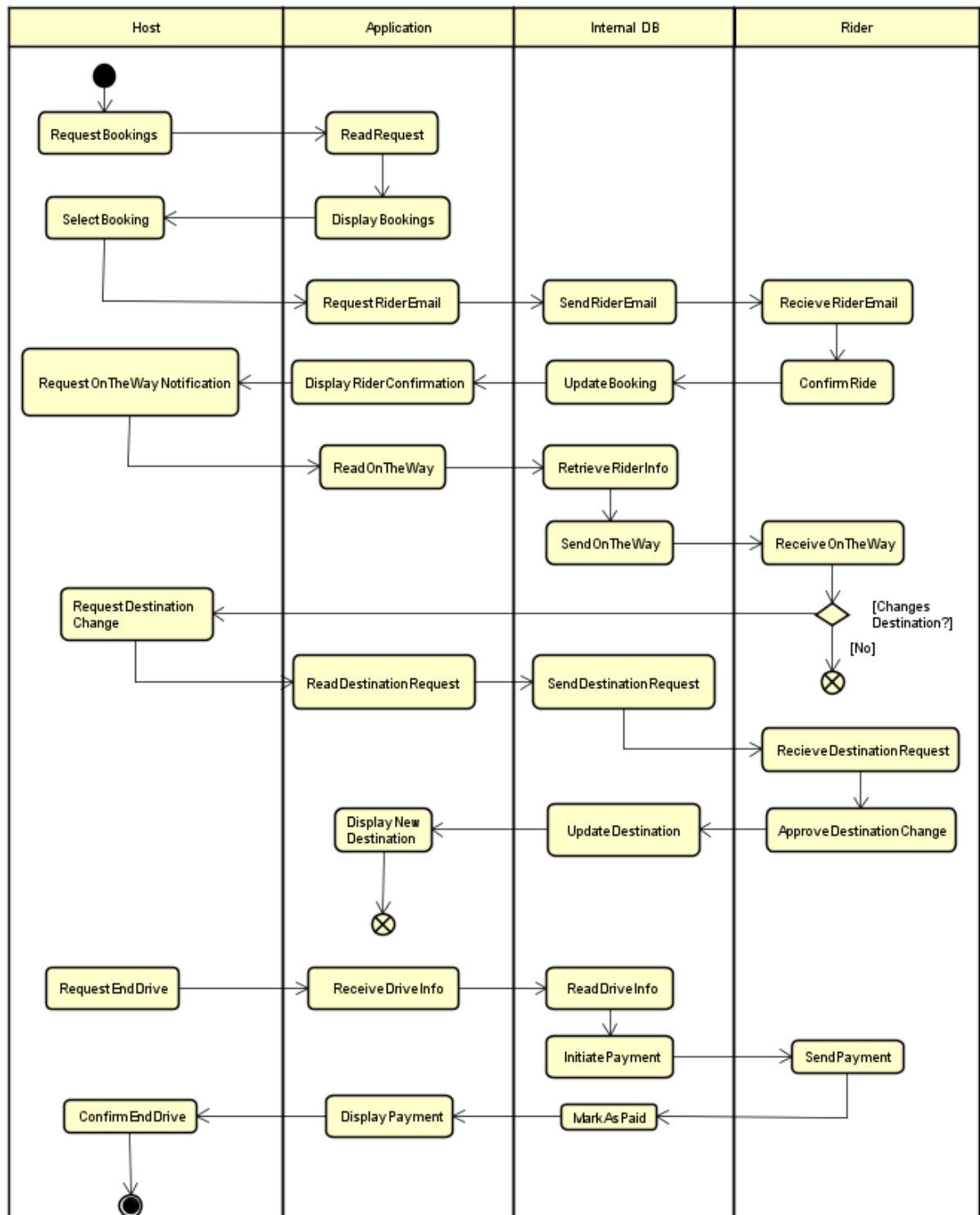


actAD3 - Login



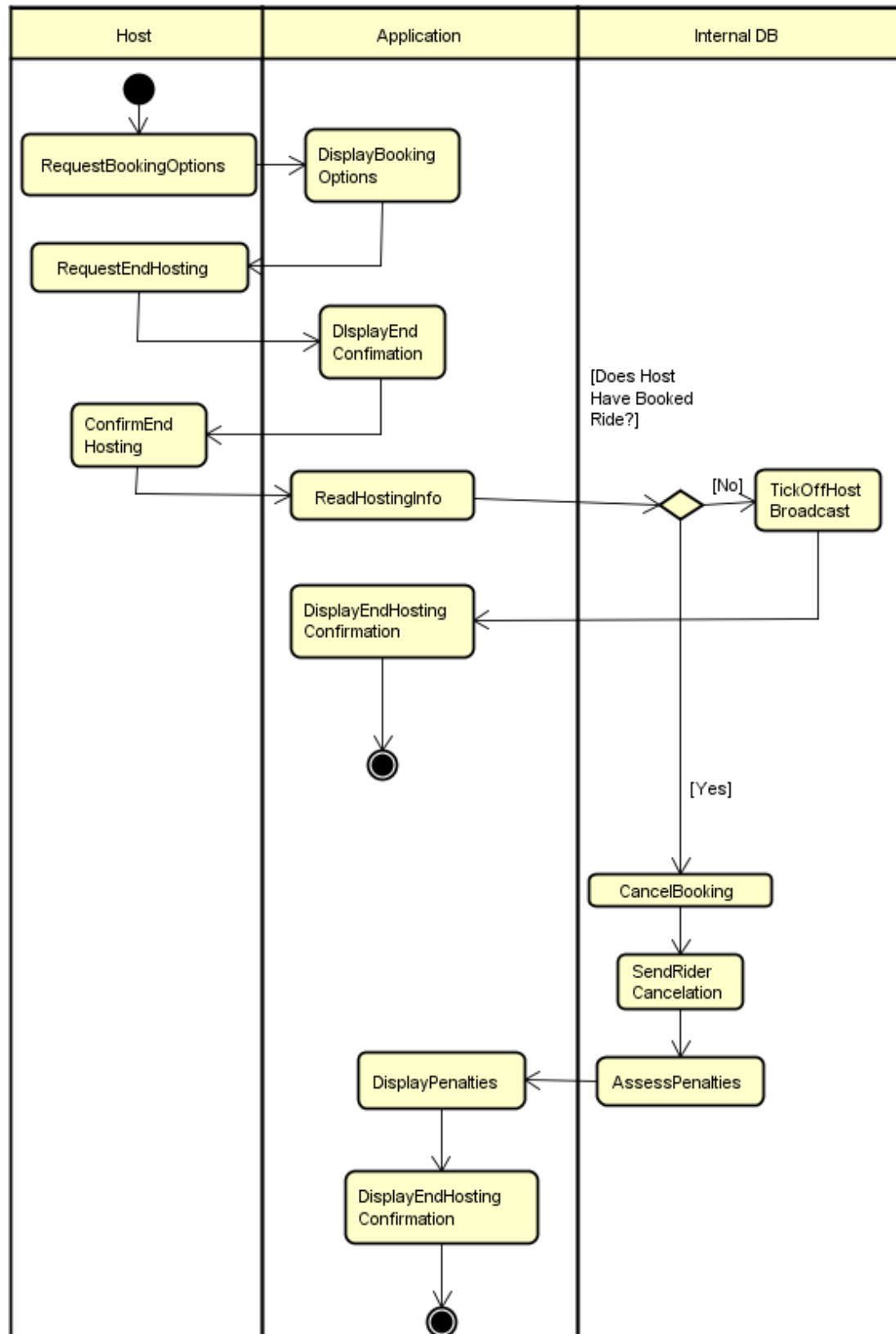
actAD4 - Account Deletion



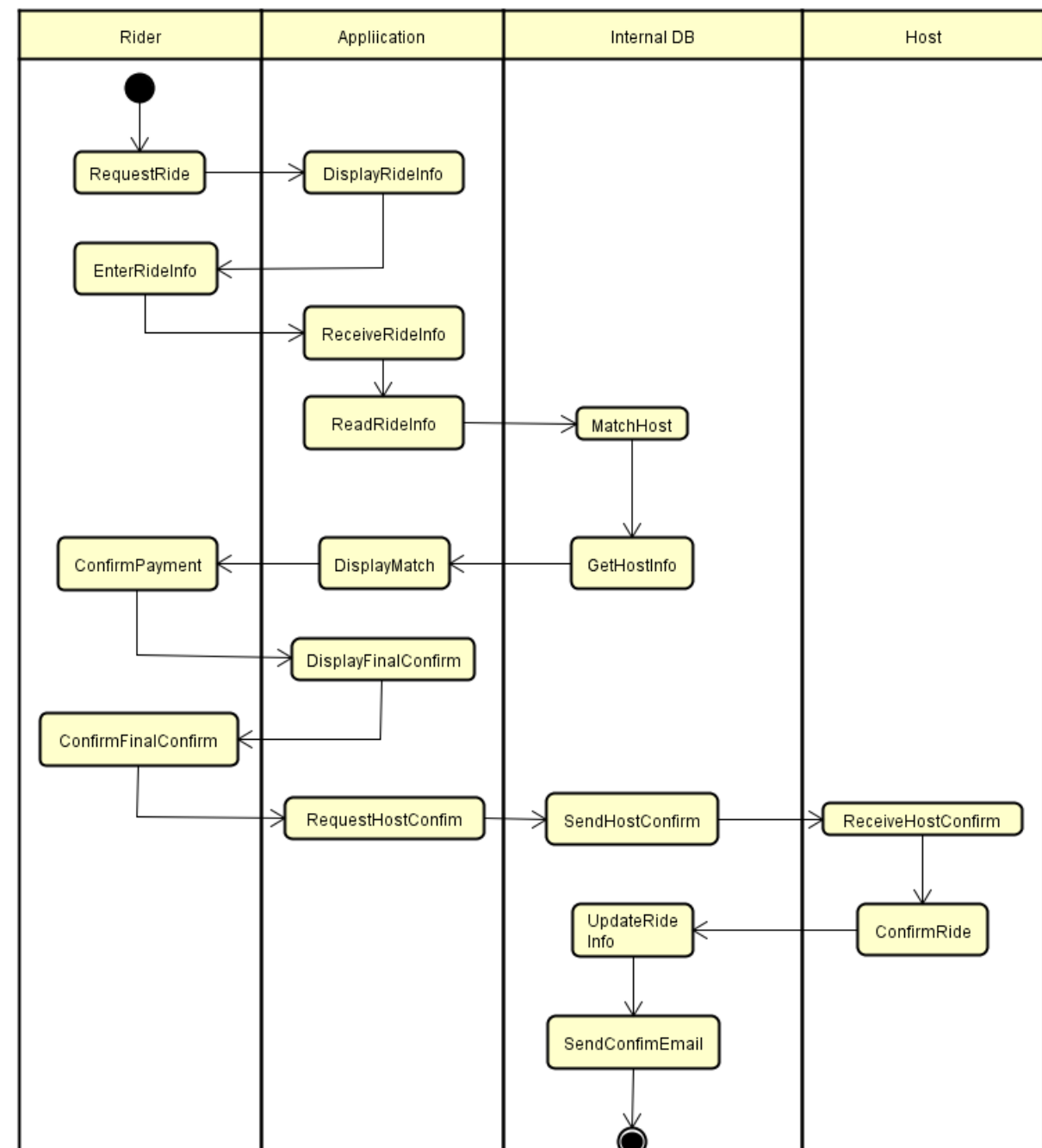




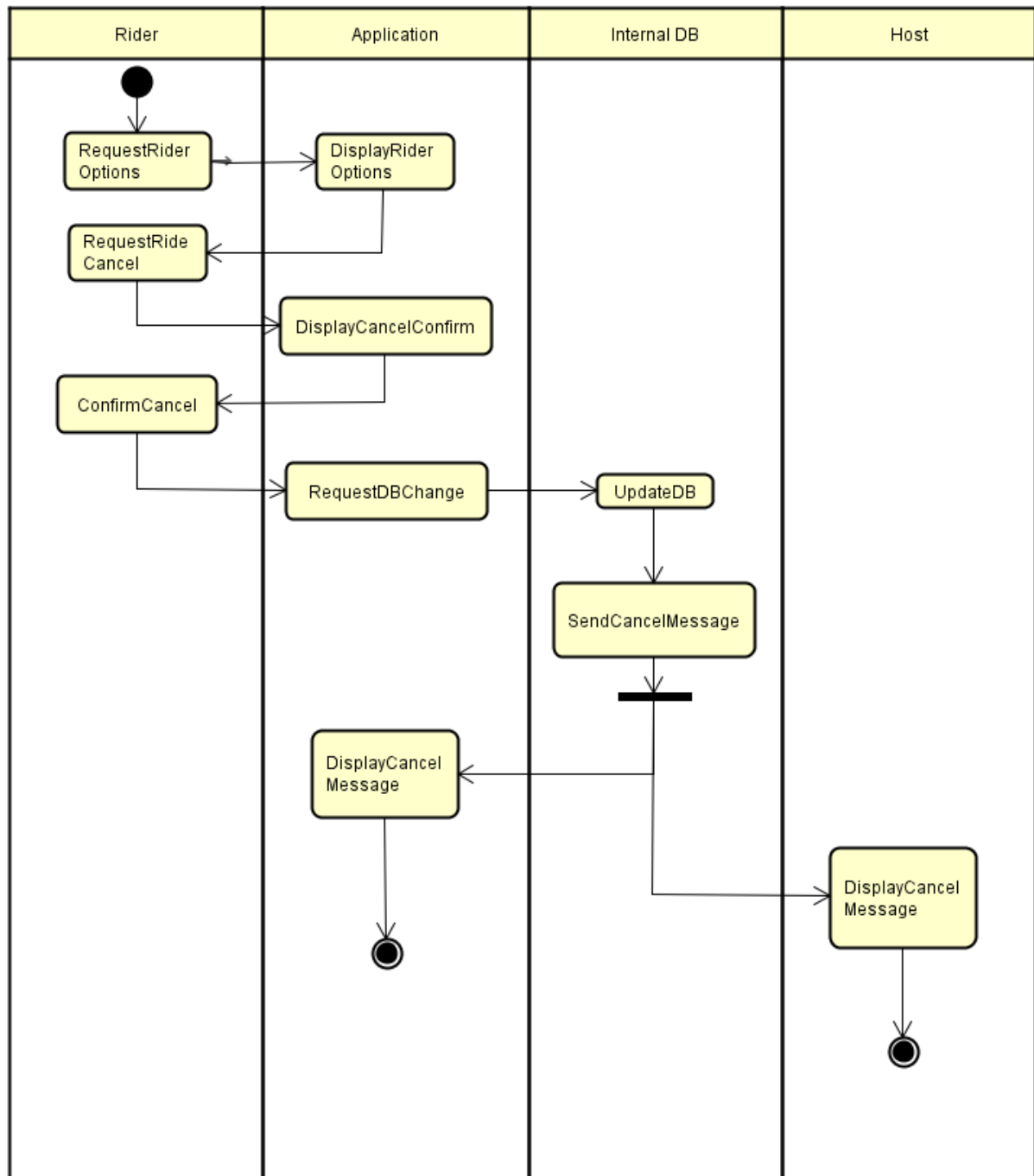
actAD6 - Removing a Hosted Ride



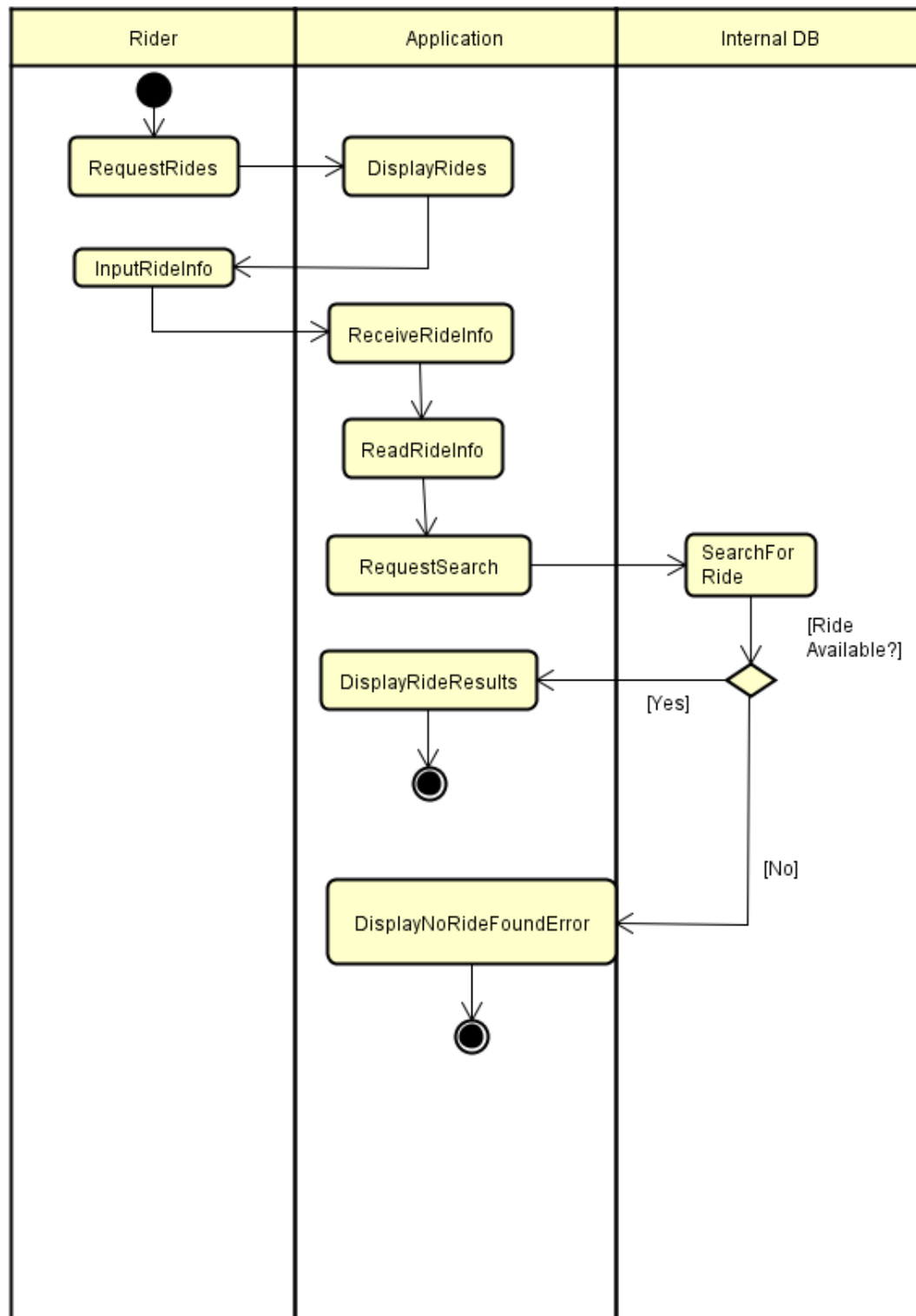
actAD7 - Requesting a Ride



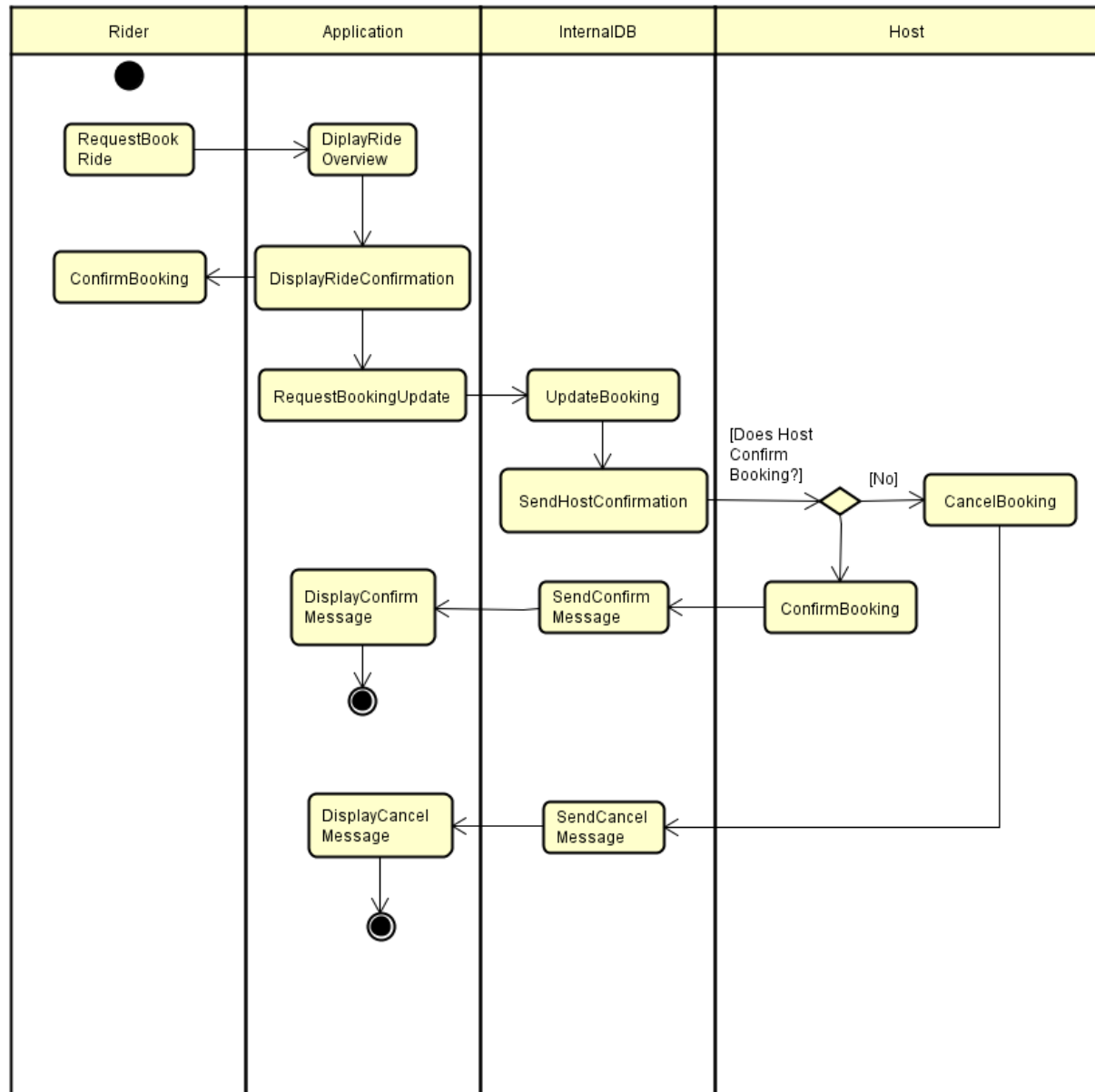
actAD8 - Cancelling a Ride as Rider



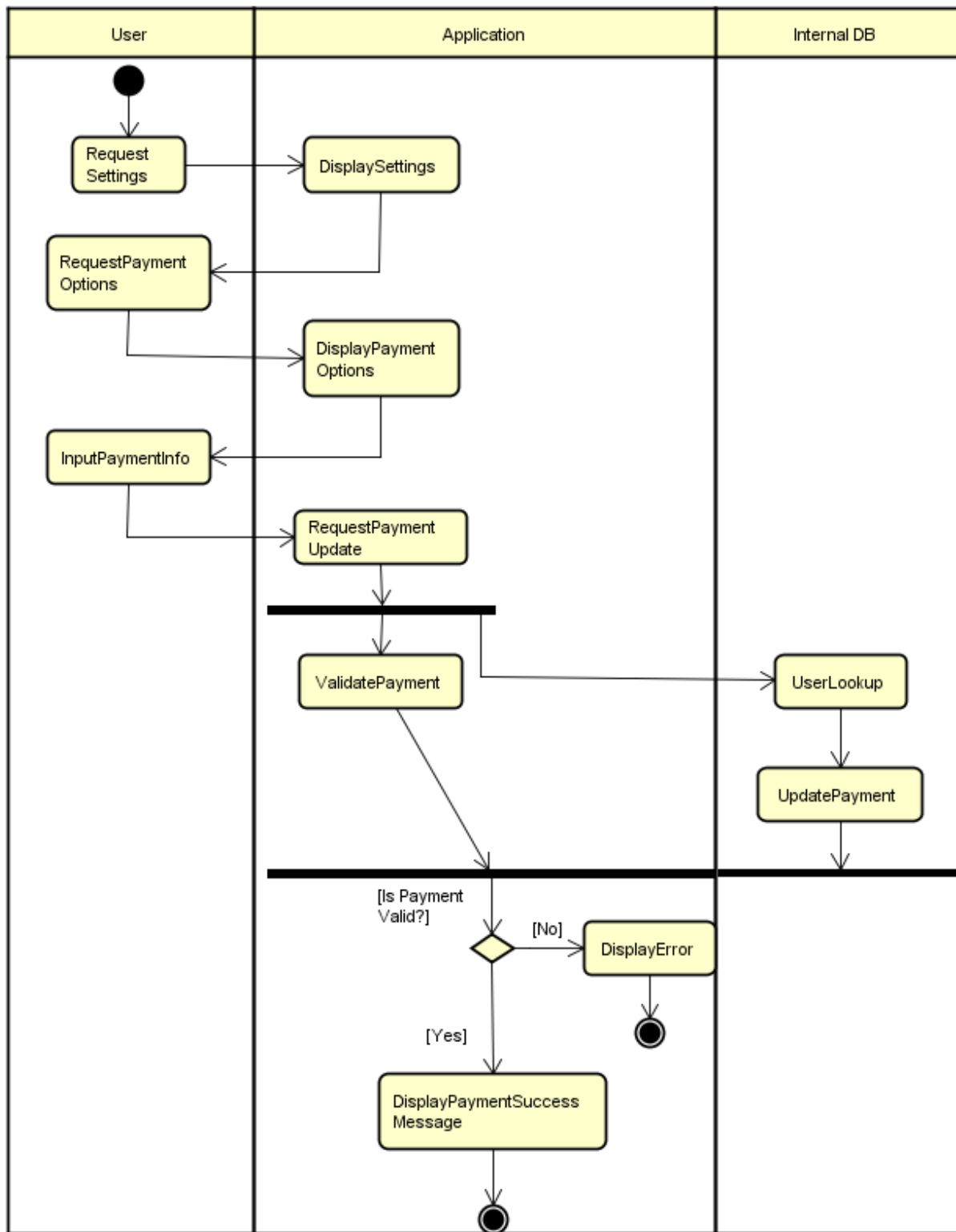
actAD9 - Browsing for a ride



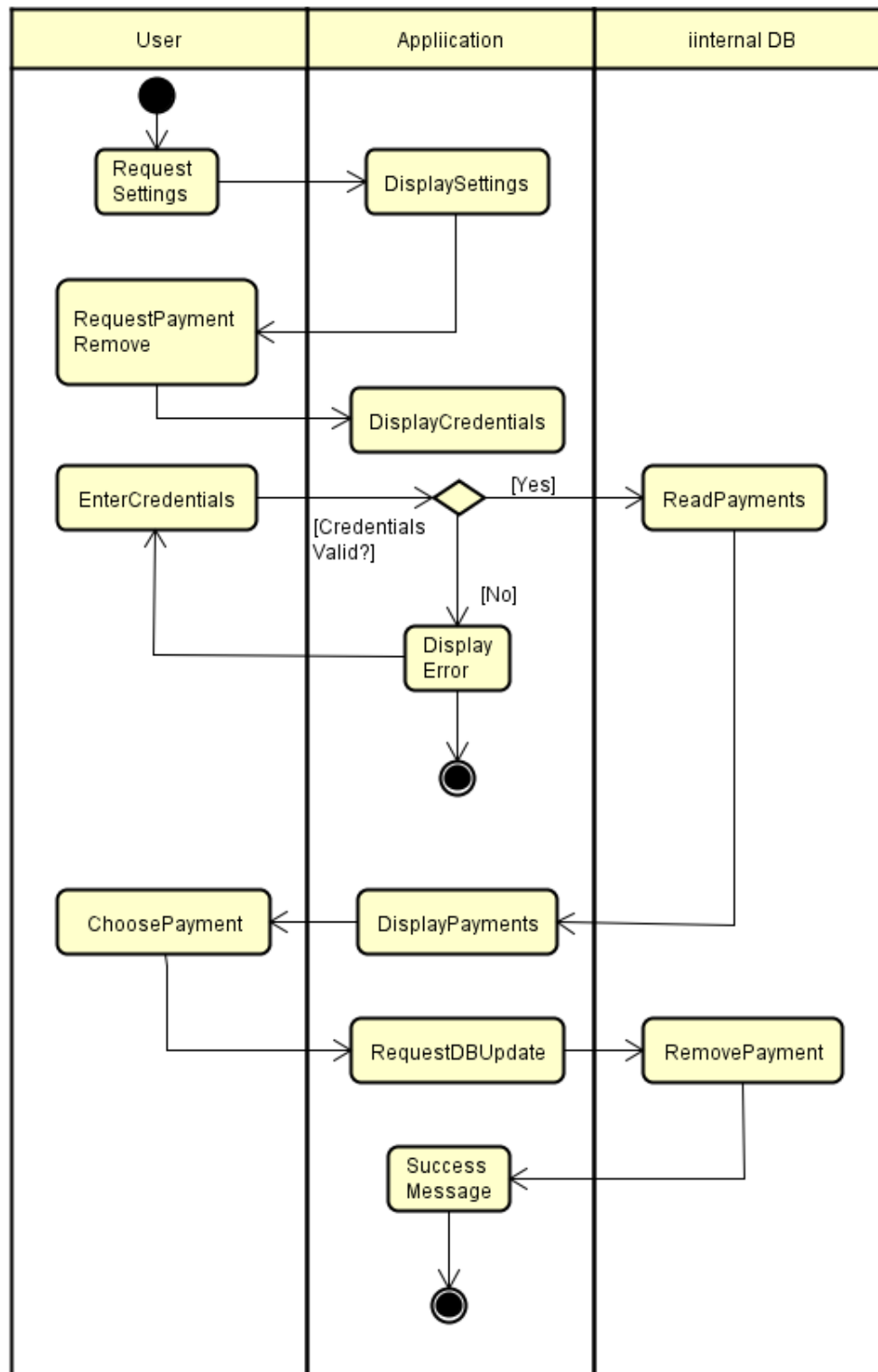
actAD10 - Joining a Ride



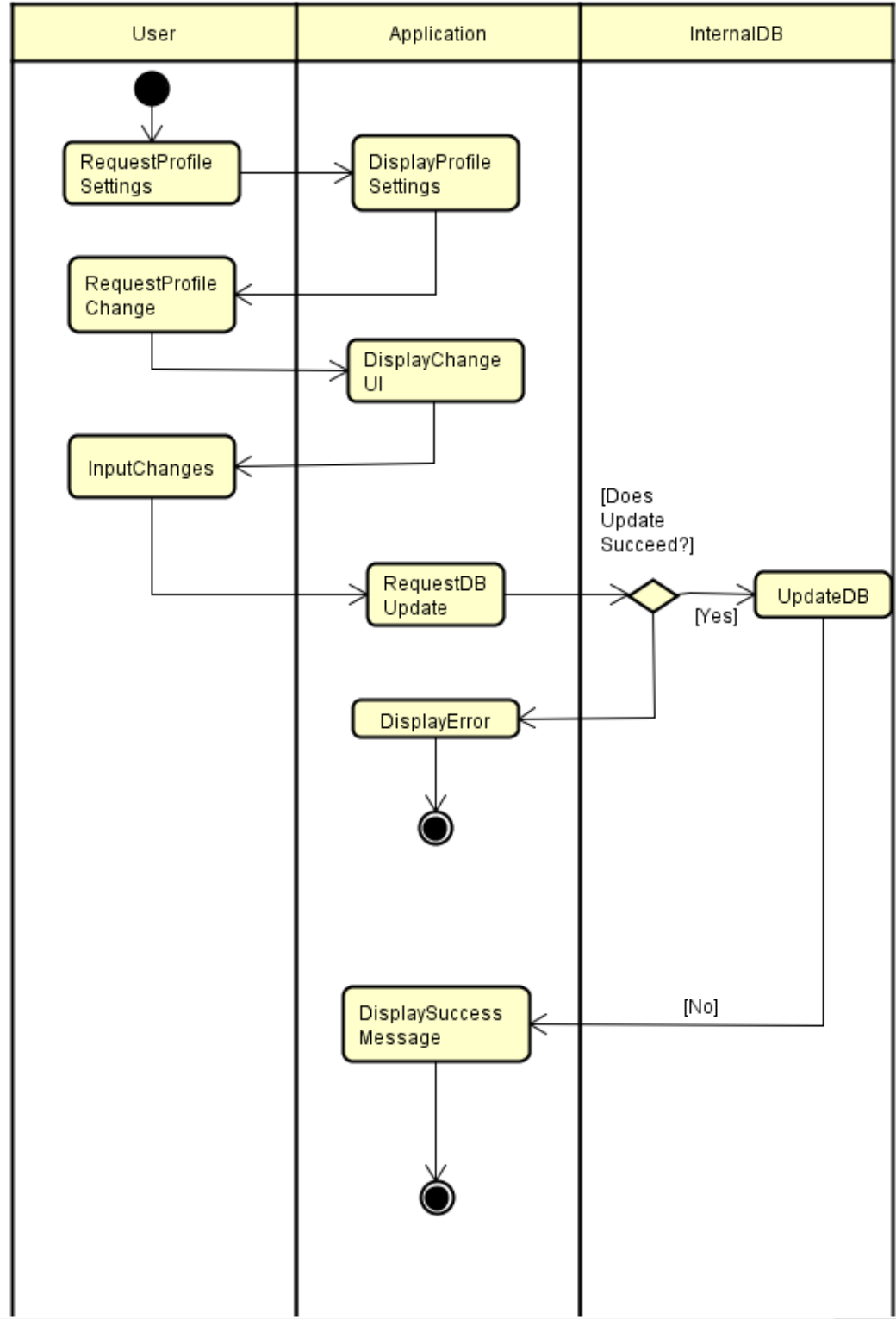
actAD11 - Adding Payment Method



actAD12 - Removing Payment Method

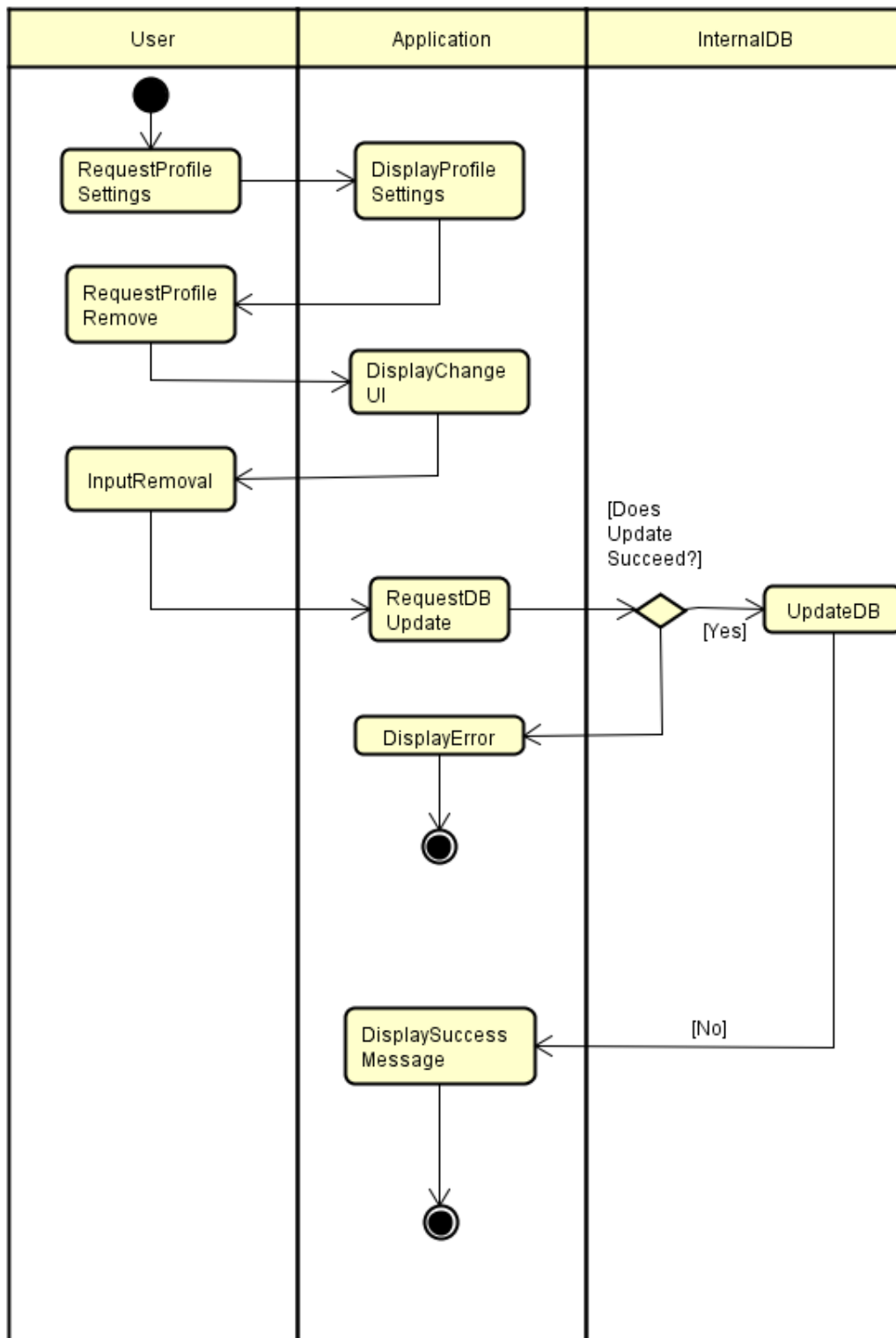


actAD13 - Adding Profile Detail

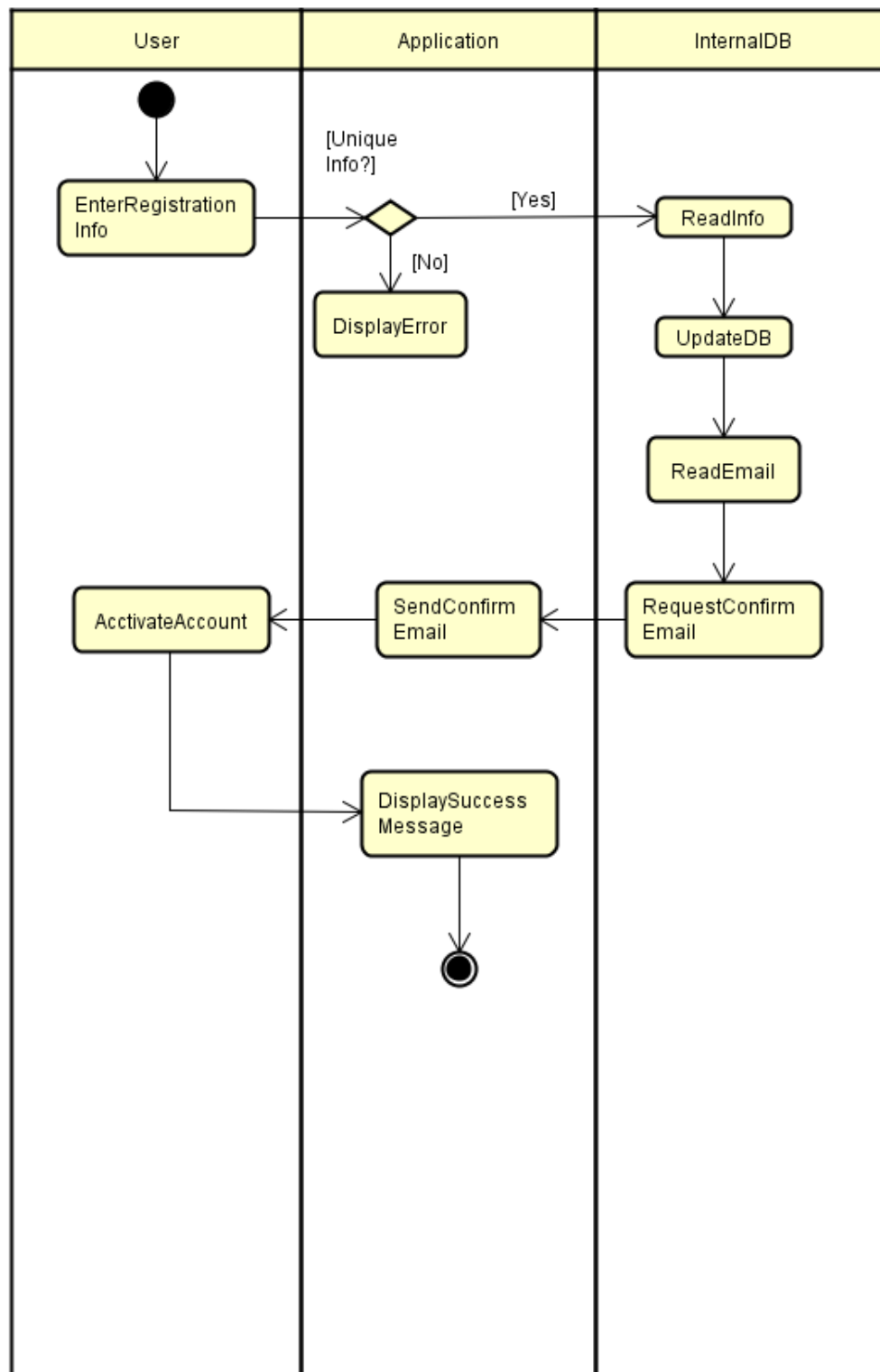




actAD14 - Removing Profile Detail



actAD15 - Receiving an Email



5. User Interface Specifications

## 5.1. Preliminary Design

Moove  
*Innovate the World*

Sign UpLog In

Register now for free &  
Share your ride

Email Address

By clicking Sign up, I agree to the Terms of Service and Privacy Policy.

Sign Up  
or  
Log In

Moove is a ride-sharing application where the driver can host the ride and the passenger(s) can join the host and share the ride.

How Moove works?


Sign Up/Log In

↓

Select the role which suits the best

Moove  
*Innovate the World*

⚙️🏠Logout



Harsh desai

Choose FileNo file chosen

Upload(Only .jpg)

Your Hosted Rides  
You haven't hosted any ride yet

Host a ride

Your Booked Rides  
You haven't booked any ride yet

Request a ride

Moove  
Innovate the World

Sign Up
Log In

Frequently Asked Questions

About Moove

What is Moove?

How does Moove work?

Who can use Moove?

What are the benefits of Moove?

Getting Started

How to sign up as a passenger or driver?

Moove

Sign Up
Log In

You Can Always Reach Us At:

1800-CALL-MOV

We are located at :

View larger map

Get Directions!

You can also check out our [FAQs](#)

Have Questions/Feedback?

Name: Full Name

Email: Email Address

Message:

Submit

About Us
Contact Us
FAQs

Log In
Sign Up
Site Map

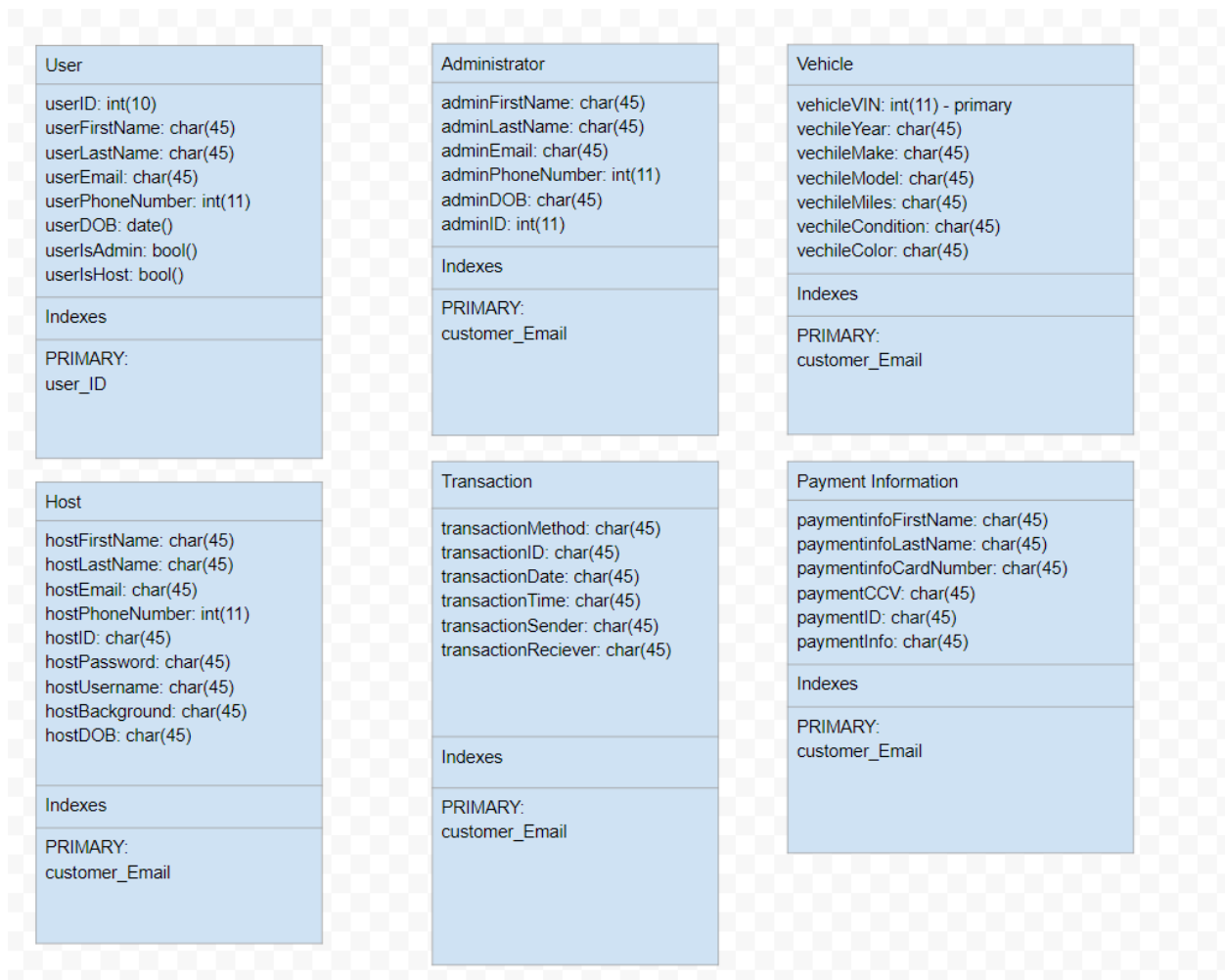
Copyright © 2018 Moove

## 5.2. User Effort Estimation

- We estimate it will take a team of 5 people around 140-160 hours of work each, or 700-800 total hours to complete this application.

## 6. Static Design

### 6.1. Class Model



## 6.2. System Operation Contracts

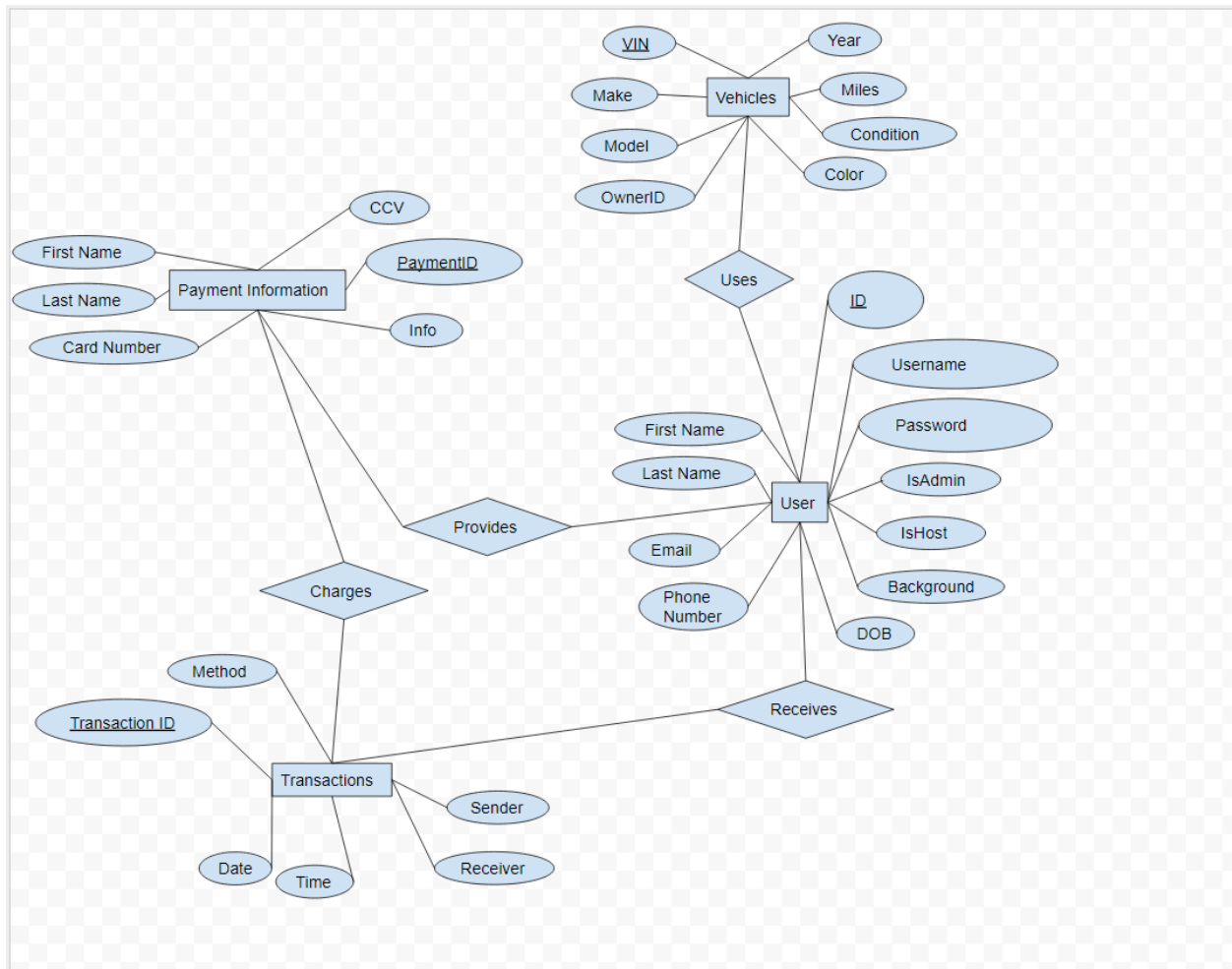
**Precondition:** Rider should have a working internet connection. Rider should open the application in the mobile app or website

**Postcondition:** Rider has successfully joined the ride that they requested.

## 6.3. Mathematical Model

$$\text{Cost} = (\text{total miles} \times .70) + 2$$

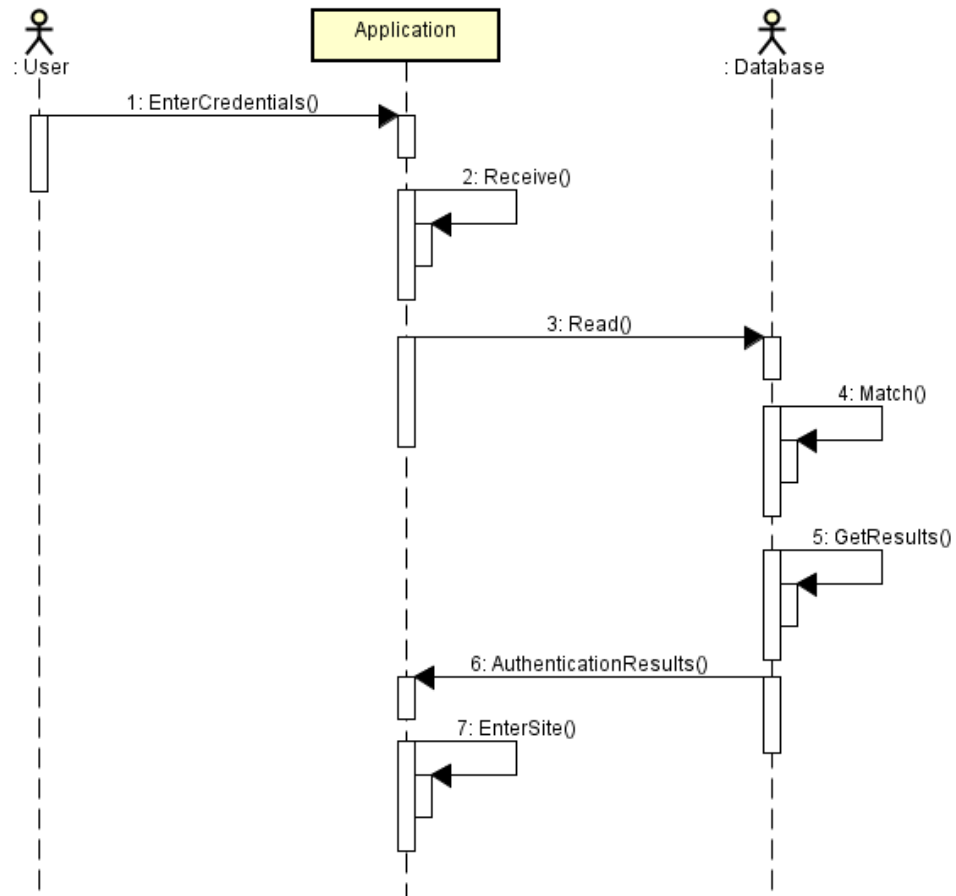
## 6.4. Entity Relation

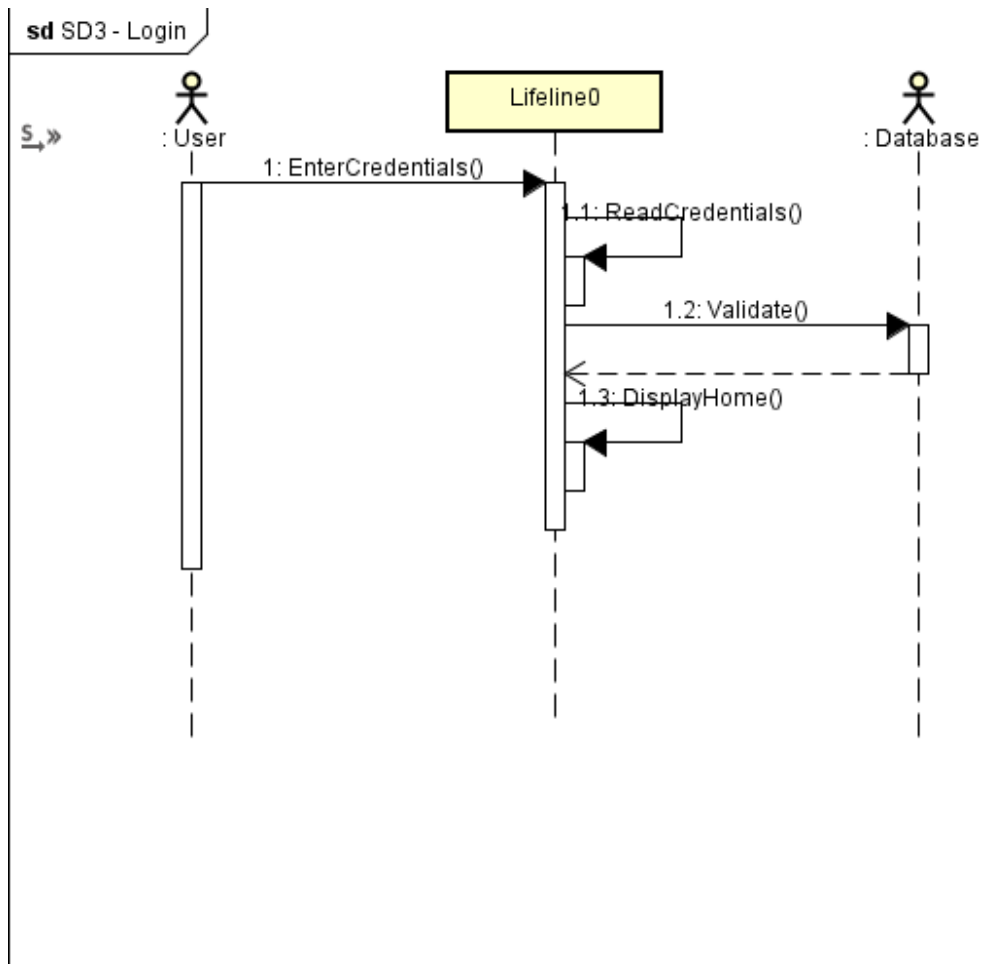


## 7. Dynamic Design

### 7.1. Sequence Diagrams.

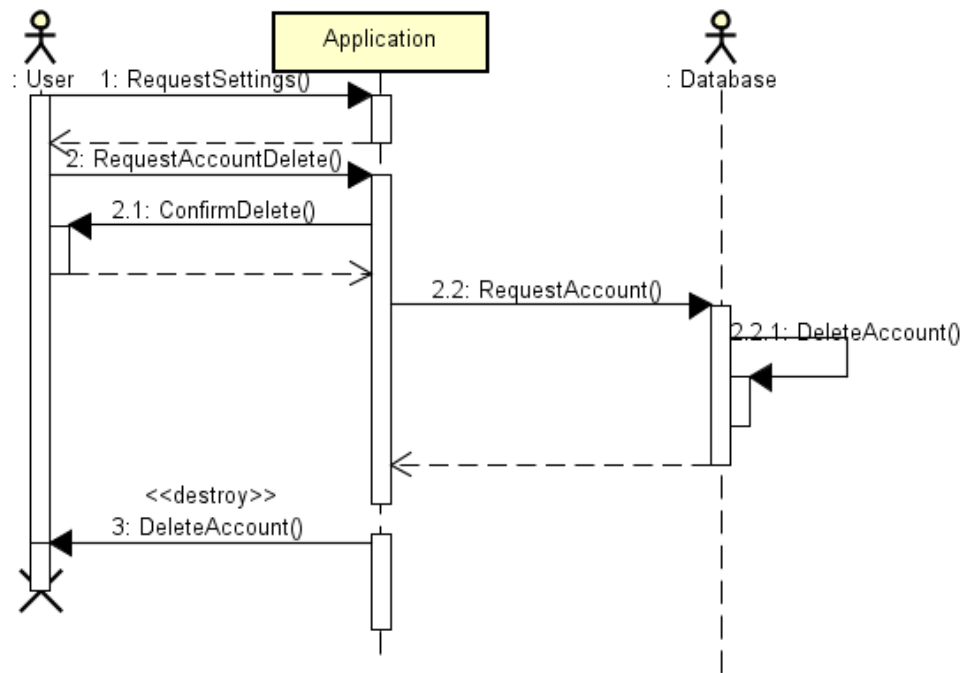
sd SD1 - Authentication



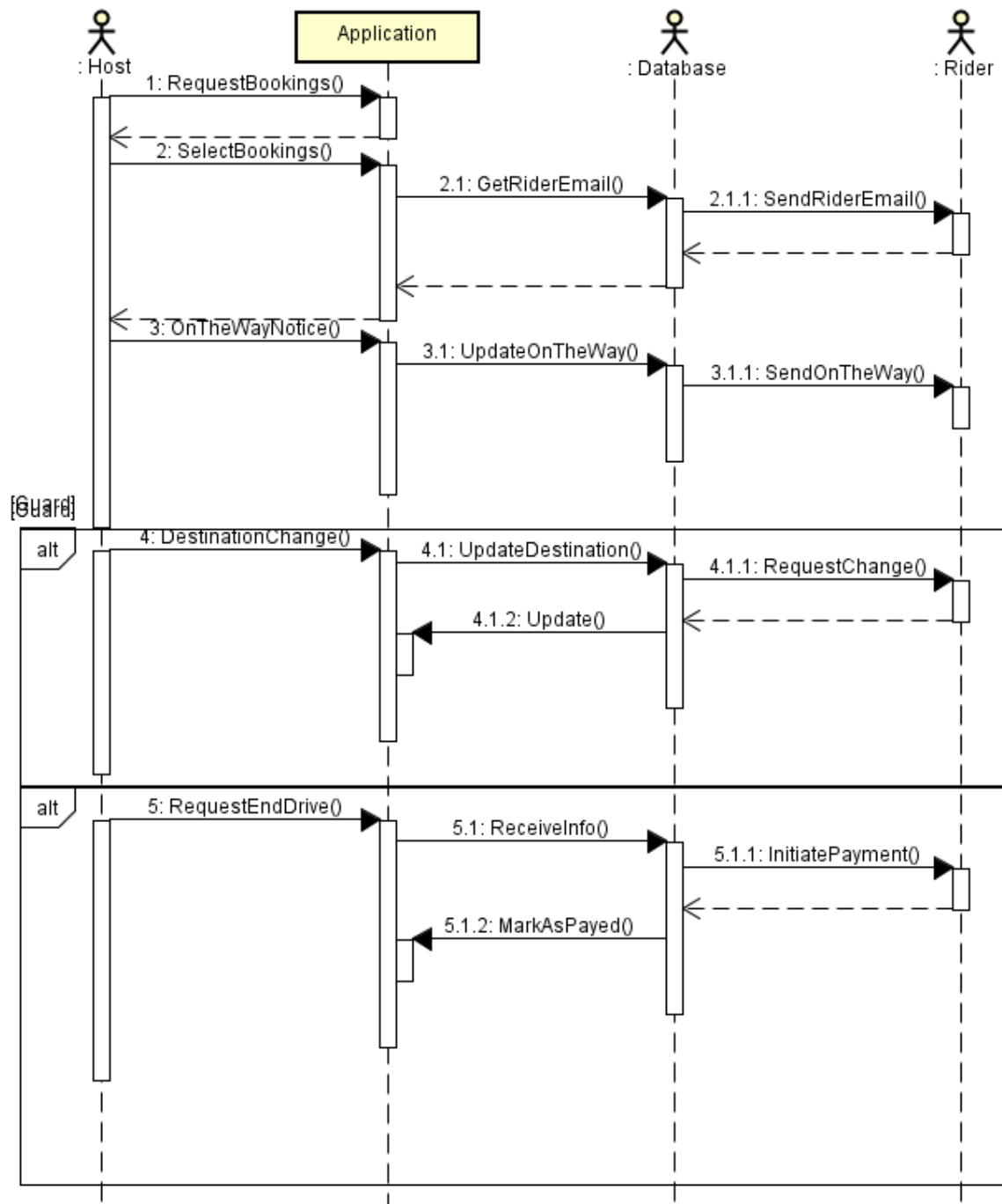




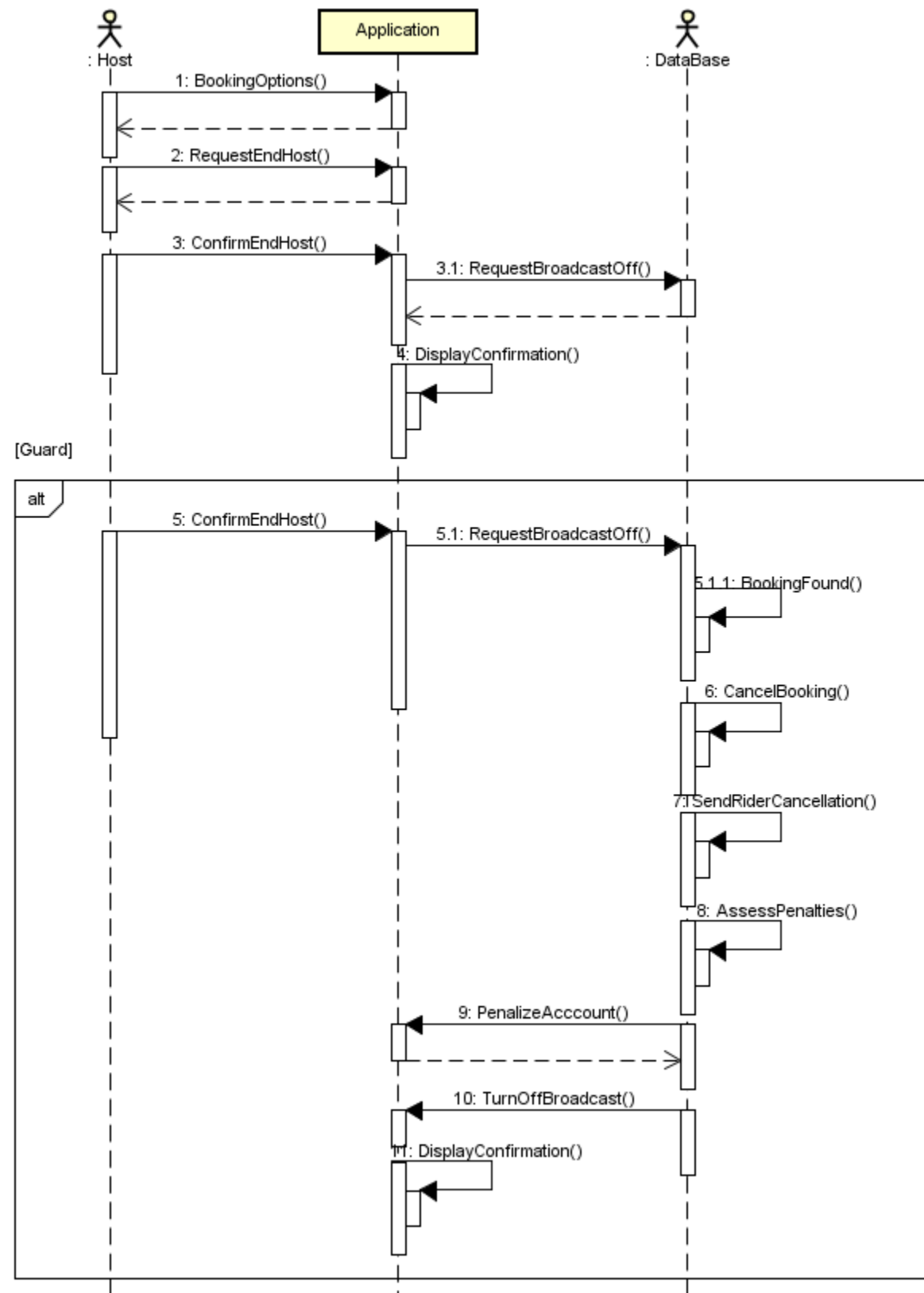
**sd** SD4 - Account Deletion



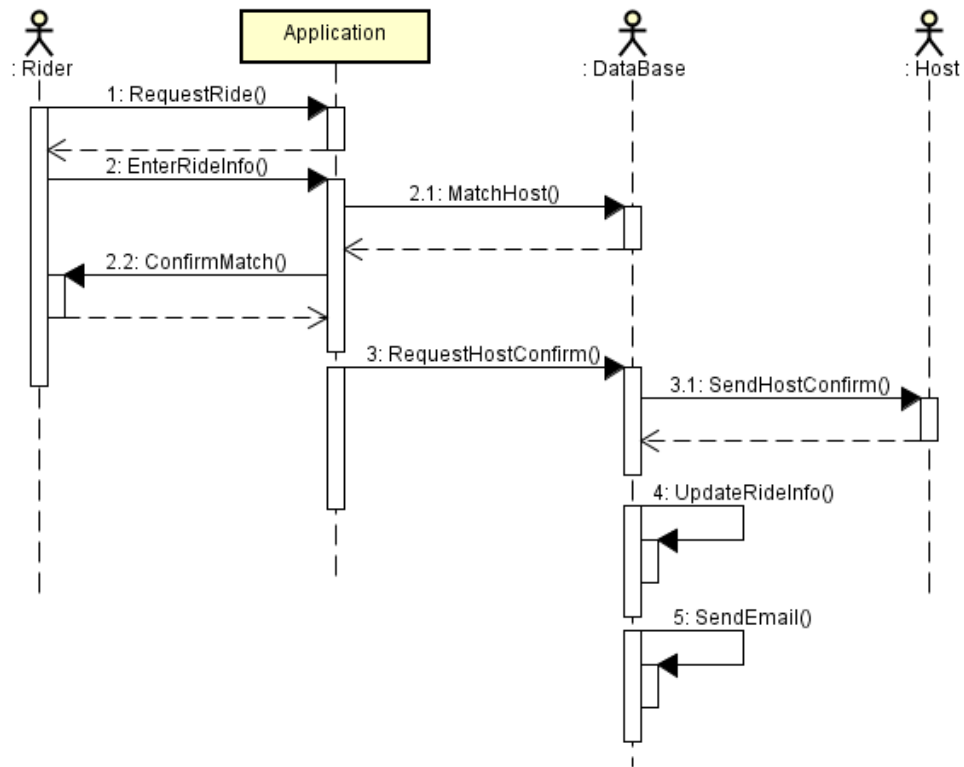
sd SD5 - Host Booking a Ride



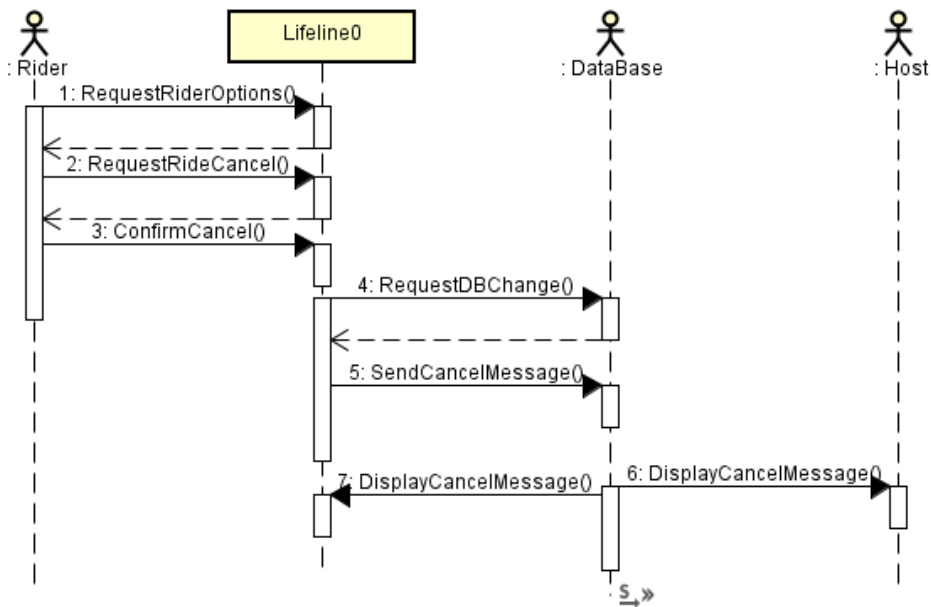
sdSD6 - Removing a Hosted Ride



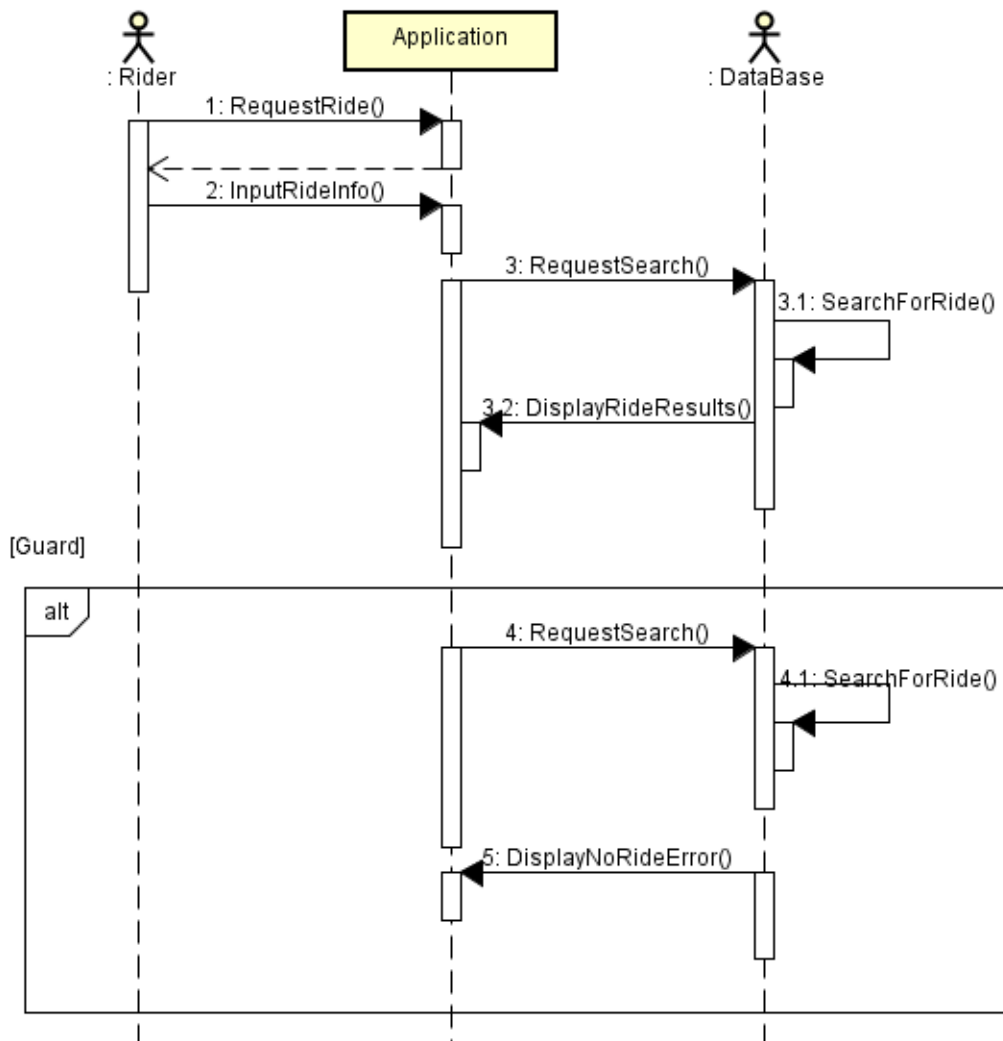
### sd SD7 - Requesting a Ride



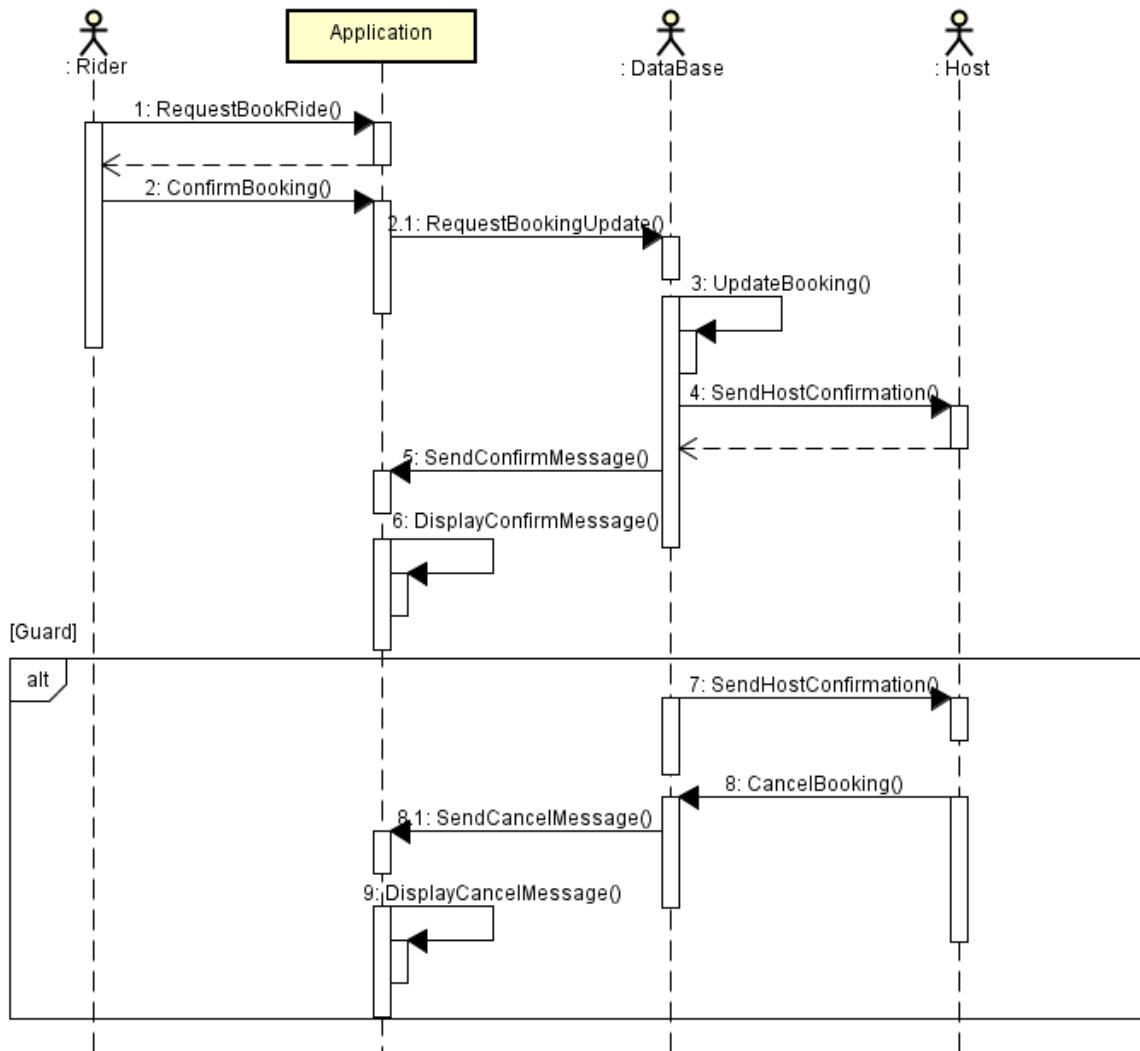
### sd SD8 - Cancelling a Ride as Rider



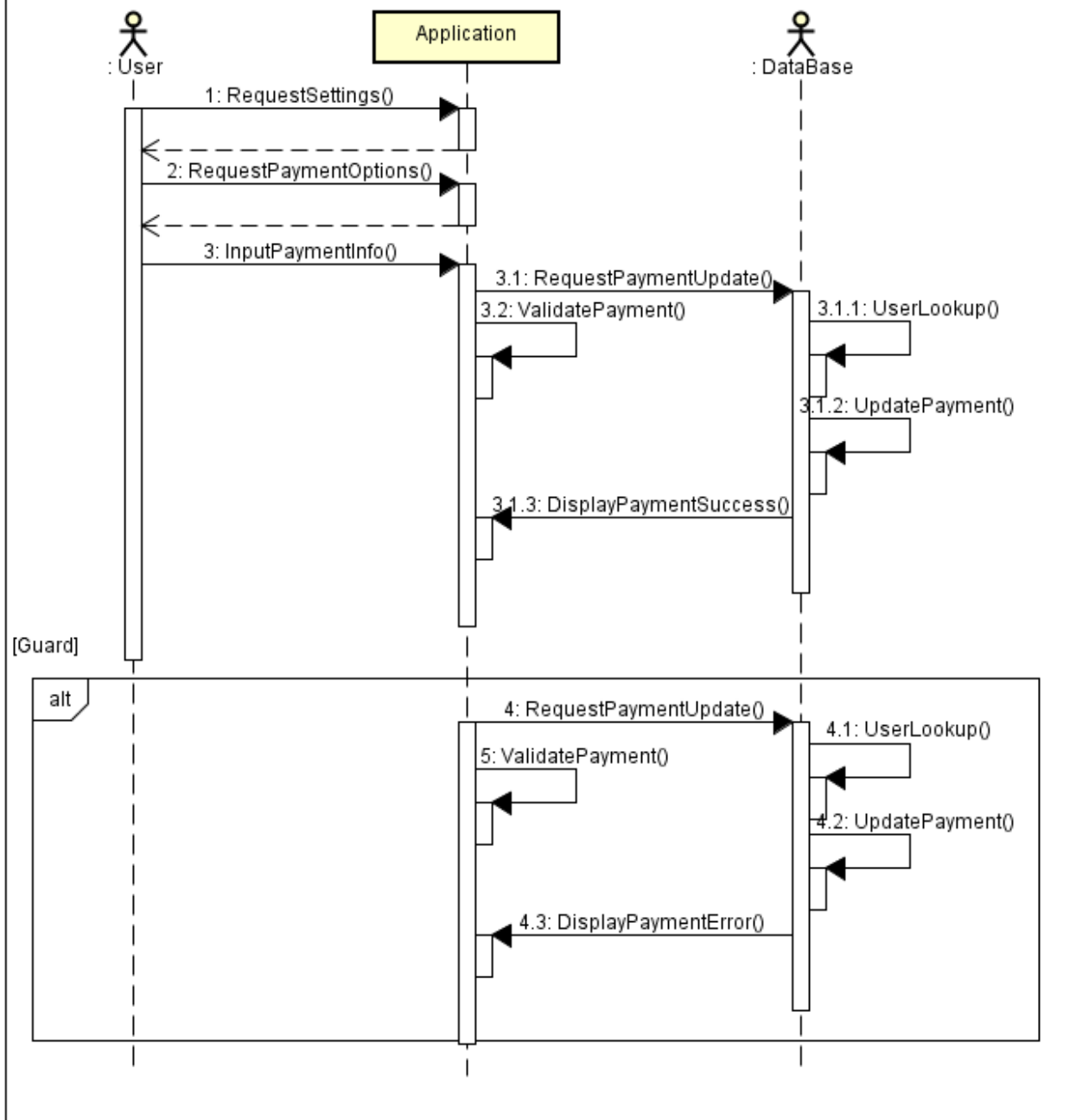
**sd** SD9 - Browsing for a Ride



sd SD10 - Joining a Ride

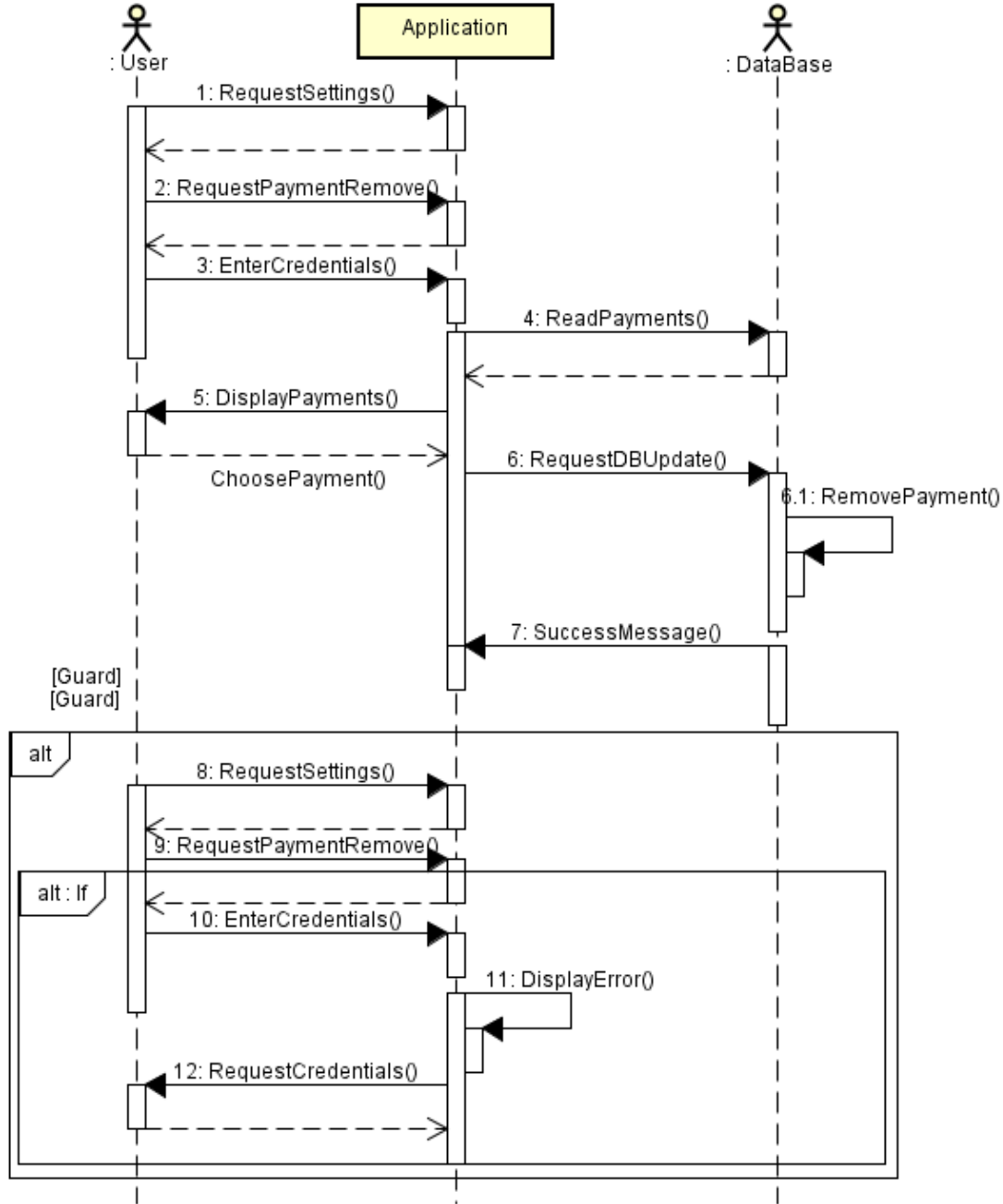


sd SD11 - Adding Payment Method



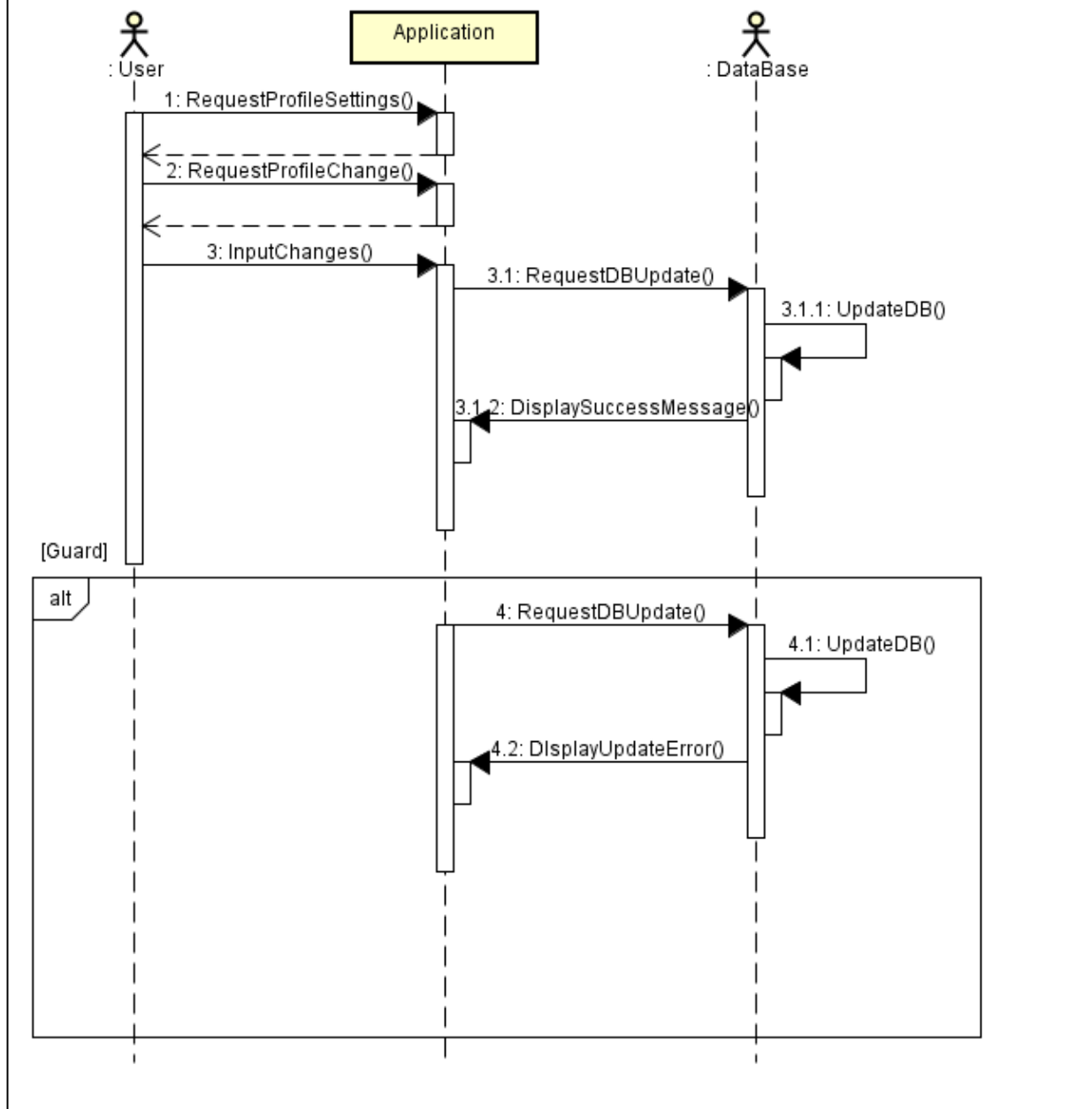
```
sequenceDiagram
    actor User as :User
    participant Application
    participant DataBase as : DataBase

    User->>Application: 1: RequestSettings()
    Application-->>User: 
    User->>Application: 2: RequestPaymentRemove()
    Application-->>User: 
    User->>Application: 3: EnterCredentials()
    Application->>DataBase: 4: ReadPayments()
    DataBase-->>Application: 
    Application->>User: 5: DisplayPayments()
    User->>Application: ChoosePayment()
    Application->>DataBase: 6: RequestDBUpdate()
    activate DataBase
    DataBase->>DataBase: 6.1: RemovePayment()
    deactivate DataBase
    DataBase-->>Application: 7: SuccessMessage()
    Application->>User: 
    alt Guard
    alt Guard
    alt
        Application->>User: 8: RequestSettings()
        Application-->>User: 9: RequestPaymentRemove()
        alt If
            Application->>User: 10: EnterCredentials()
            Application->>Application: 11: DisplayError()
            Application->>User: 12: RequestCredentials()
            Application-->>Application: 
        else
        else
        else
    end
    end
    end
```

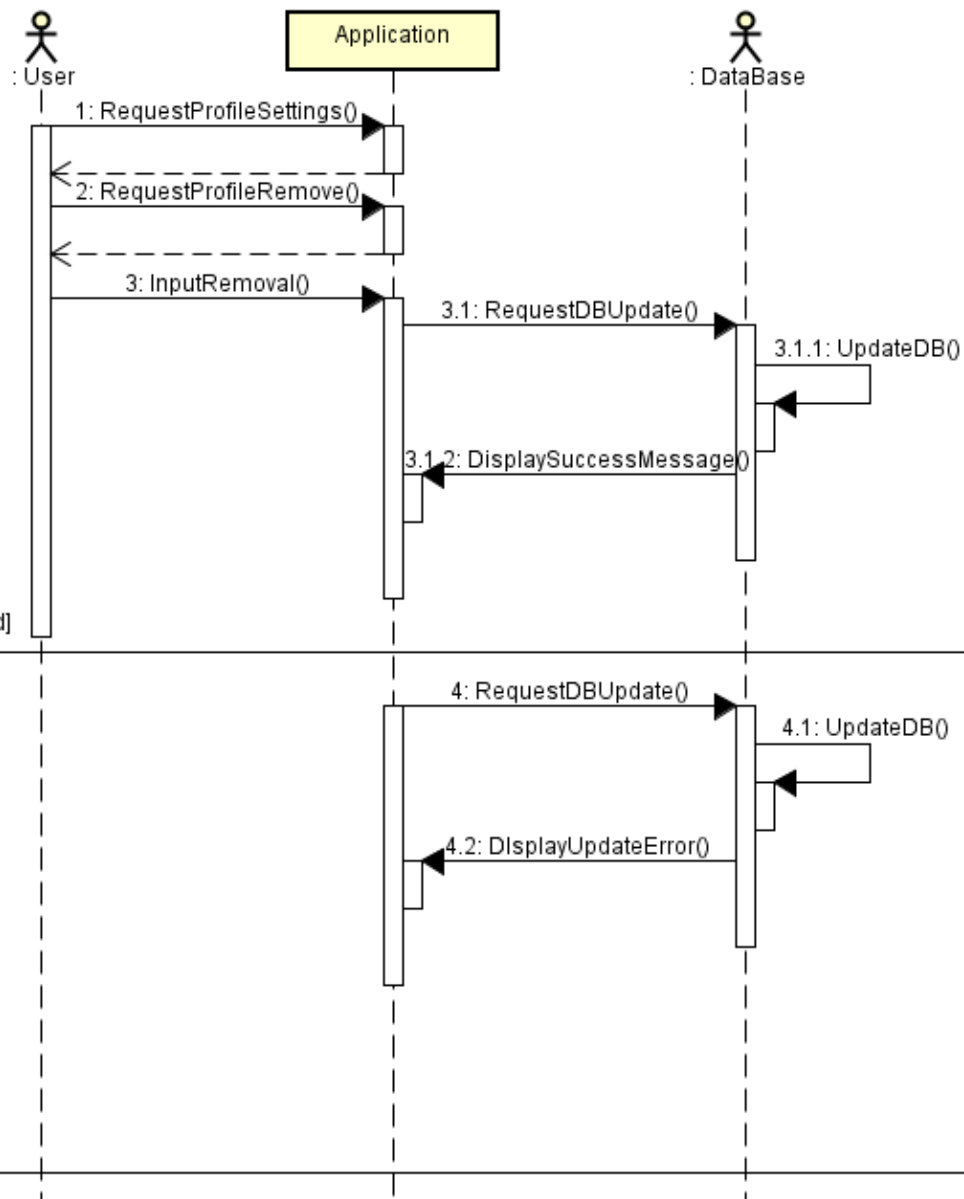


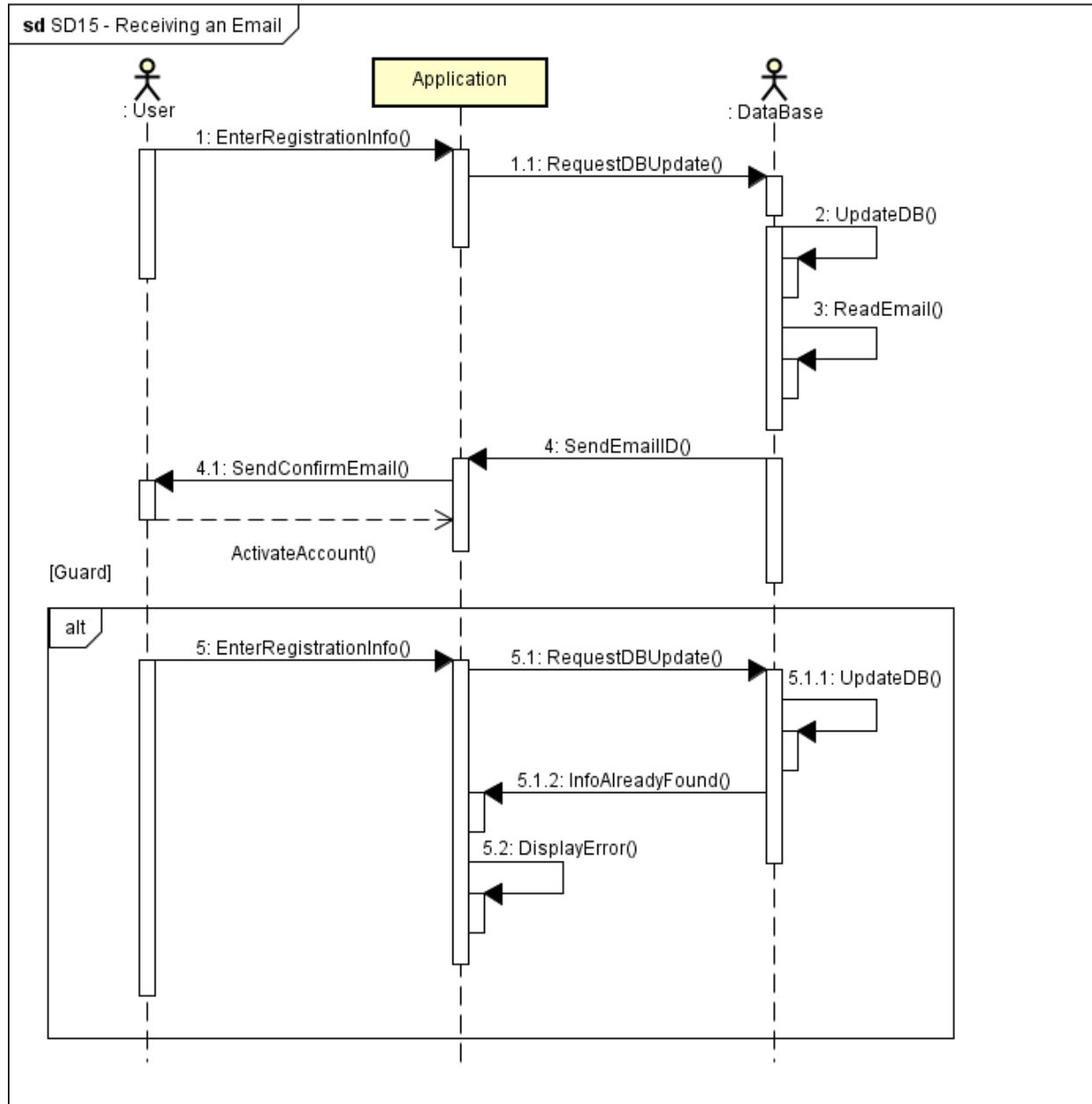


**sd** SD13 - Adding Profile Detail



sd SD14 - Removing Profile Detail

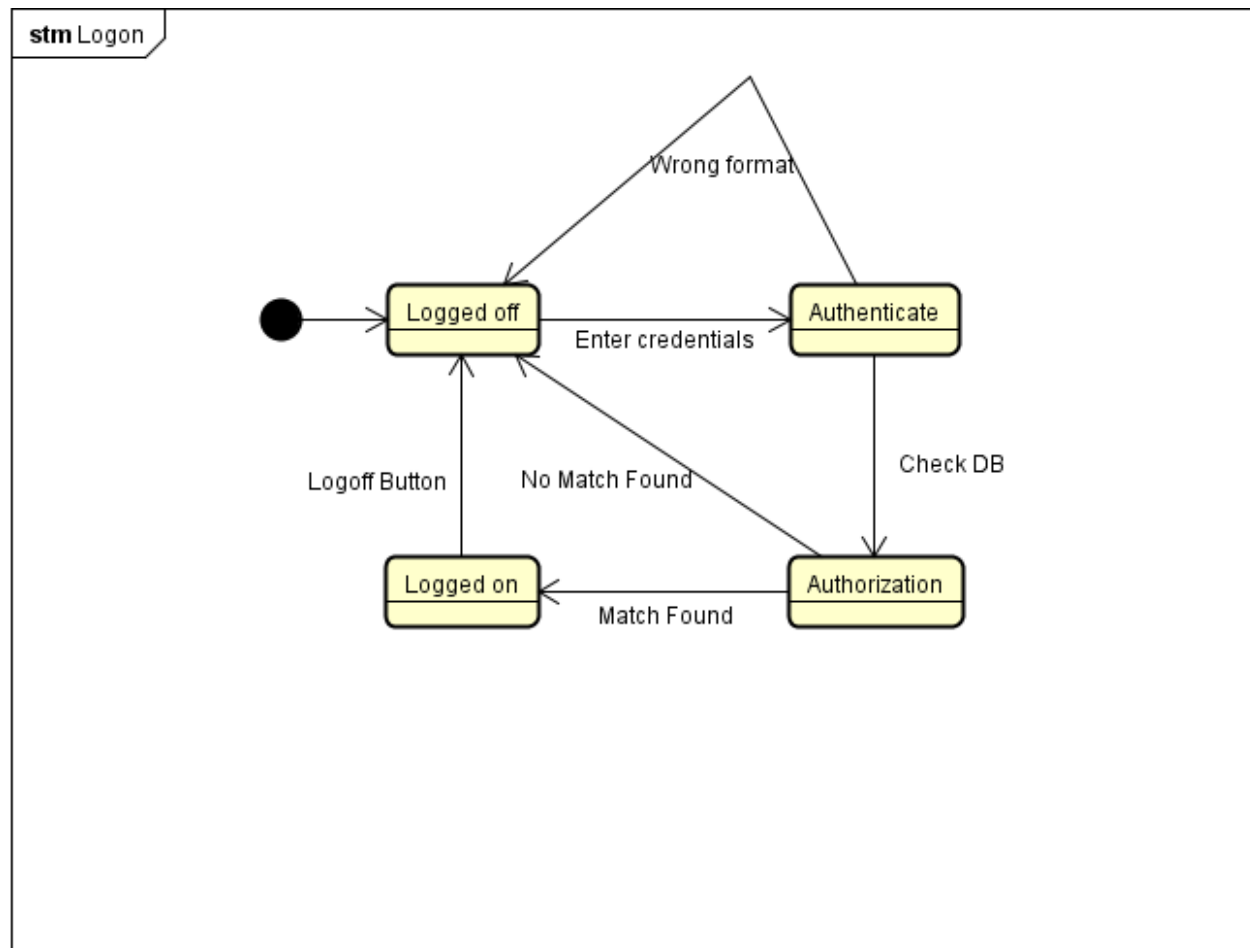
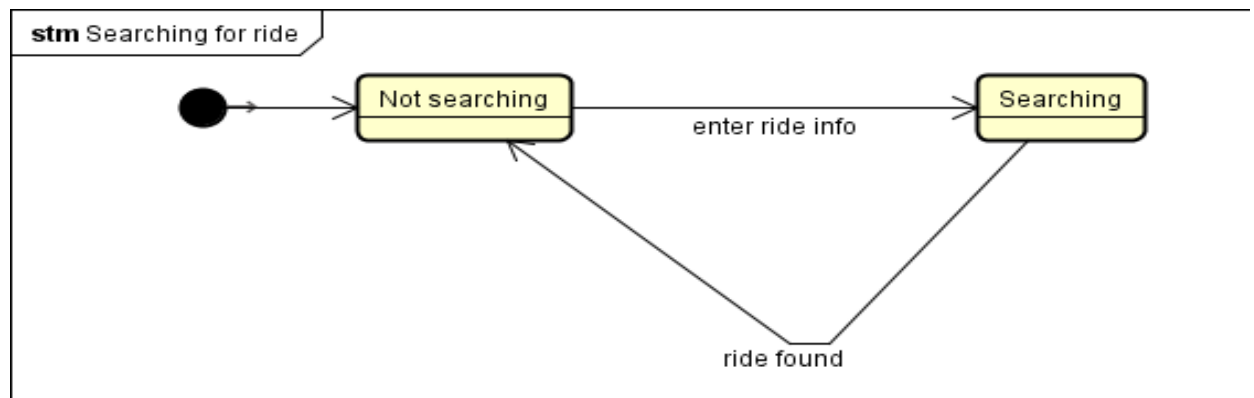




## 7.2. Interface Specification

- **Hosted Ride:** This is where the user is able to host a ride.
- **Requested Ride:** This is where user can request a ride.
- **Browse Ride:** Users are able to see what rides are available nearby and they can view my hosted ride and my requested ride.
- **Profile:** Allows the users to edit and add their personal information.

## 7.3. State Diagrams



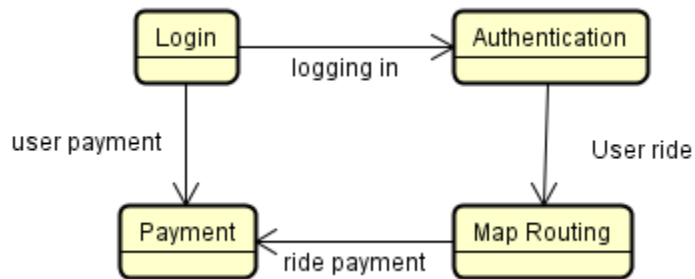
## 8. System Architecture and System Design

### 8.1. Subsystems / Component / Design Pattern Identification

- Subsystems
  - Login
  - Authentication

- Payments
  - Map routing
- Components
  - Login form
  - Request ride form
  - Host ride form
  - Stripe payment form
  - Profile form
  - Map location auto fill
- Design Pattern Identification
  - Bootstrap template
  - Popup form screens

## 8.2. Mapping Subsystems to Hardware Deployment Diagram)



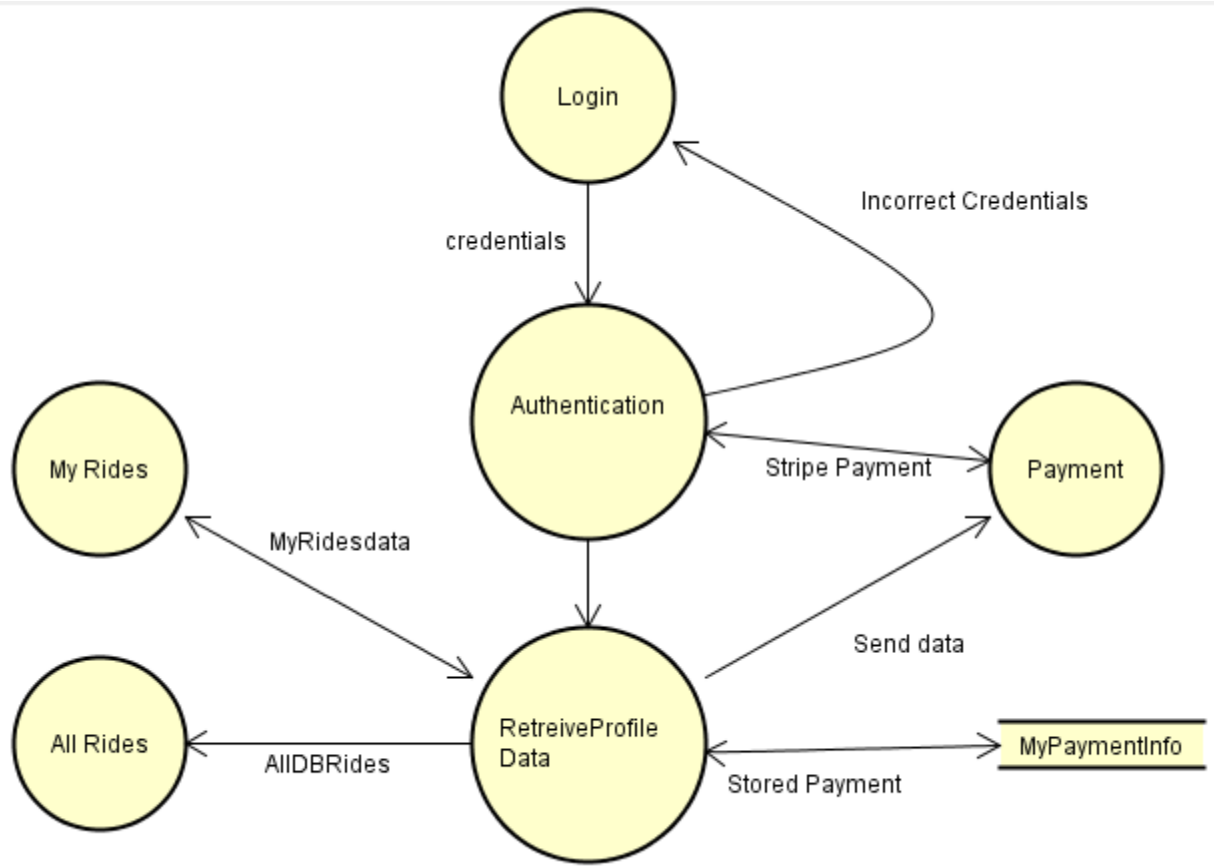
## 8.3. Persistent Data Storage

- MySQL database
- Stripe Payment info database

## 8.4. Network Protocol

- HTTP
- HTTPS

## 8.5. Global Control Flow



#### 8.6. Hardware Requirement

- Any computer or mobile device that has internet access and can run any web browser.

## 9. Algorithms and Data Structures

### 9.1. Algorithms

- Google Waypoint Algorithm

```

<?php } ?>
directionsService.route({
    origin: document.getElementById('start').value,
    destination: document.getElementById('end').value,
    waypoints: waypoints,
    optimizeWaypoints: true,
    travelMode: 'DRIVING'
}, function(response, status) {
    if (status === 'OK') {
        directionsDisplay.setDirections(response);
        var route = response.routes[0];
        var summaryPanel = document.getElementById('directions-panel');
        summaryPanel.innerHTML = '';
        // For each route, display summary information.
        for (var i = 0; i < route.legs.length; i++) {
            var routeSegment = i + 1;
            summaryPanel.innerHTML += '<b>Route Segment: ' + routeSegment +
                '</b><br>';
            summaryPanel.innerHTML += route.legs[i].start_address + ' to ';
            summaryPanel.innerHTML += route.legs[i].end_address + '<br>';
            summaryPanel.innerHTML += route.legs[i].distance.text + '<br><br>';
        }
    }
});
}

```

## 2. Distance Matrix Algorithm

```

function callback(response, status) {
    if (status != google.maps.DistanceMatrixStatus.OK) {
        $('#hostrideresult').html(err);
    } else {
        var origin = response.originAddresses[0];
        var destination = response.destinationAddresses[0];
        if (response.rows[0].elements[0].status === "ZERO_RESULTS") {
            $('#hostrideresult').html("Better get on a plane. There are no roads between "
                + origin + " and " + destination);
        } else {
            var distance = response.rows[0].elements[0].distance;
            var distance_value = distance.value;
            var distance_text = distance.text;
            var miles = distance_text.replace(","," ", true);
            //var miles = distance_text.substring(0, distance_text.length - 3);
            miles = miles.replace("mi"," ", true);
            var pay = miles * 0.4;

            $('#hostrideresult').html("It is " + distance_text + " miles from " + origin + " to " + destination + "<br/>Total earning per passenger is: <strong>"
                + pay + "</strong>");
        }
    }
}

String.prototype.replaceAll = function(str1, str2, ignore) {
    return this.replace(new RegExp(str1.replace(/[^\w\s\[\]\/\|\,.\*\*\+\-\?\\\{\}\~\!'\"<\/script>

```

## 9.2. Data Structures

The usage of data structures in the project consisted of:

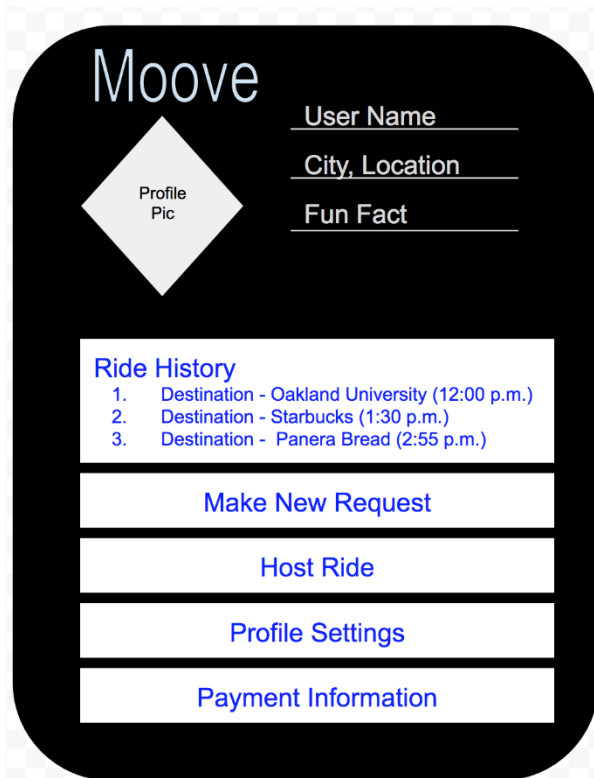
### 1. Arrays (Single Dimensional)

```
function calculateAndDisplayRoute(directionsService, directionsDisplay) {
    var waypoints = [];
    <?php
    for($i=0;$i<count($waypoint);$i++) { ?>
    waypoints.push({
        location: '<?php echo $waypoint[$i] ?>',
        stopover: true
    });
    <?php } ?>
    directionsService.route({
        origin: document.getElementById('start').value,
        destination: document.getElementById('end').value,
        waypoints: waypoints,
        optimizeWaypoints: true,
        travelMode: 'DRIVING'
    }, function(response, status) {
        if (status === 'OK') {
            directionsDisplay.setDirections(response);
            var route = response.routes[0];
            var summaryPanel = document.getElementById('directions-panel');
            summaryPanel.innerHTML = '';
            // For each route, display summary information.
            for (var i = 0; i < route.legs.length; i++) {
                var routeSegment = i + 1;
                summaryPanel.innerHTML += '<?php echo $routeSegment ?>';
            }
        }
    });
}
```


## 10. User Interface Design and Implementation

### 10.1. User Interface Design





# New Ride Request

current location 

Destination

ETA: ~~~~~

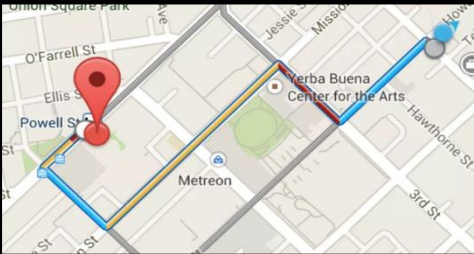
Google Map Preview

SUBMIT

# Moove

[Make New Request](#)

Current Location: 1883 Long Lake Road, West Bloomfield, MI 48323  
 Destination: 318 Meadow Brook Rd, Rochester, MI 48309  
 Estimated Time Arrival: 2:37 p.m.



[Ride History](#)

1. Destination - Oakland University (12:00 p.m.)
2. Destination - Starbucks (1:30 p.m.)
3. Destination - Panera Bread (2:55 p.m.)

[Submit](#)

logo

# MOOVE

Email

Password



## 10.2. User Interface Implementation

Moove

Innovate the World

Logout

Adam Vida

Choose File

No file chosen

Upload!Only .jpg

Host a ride

Request a ride

Browse Rides

About Us

Contact Us

FAQs

f

Copyright © 2018 Moove

Host a ride

Book a ride

Site Map

Moove

Innovate the World

Logout

Map

Satellite

Start

Hilton Auburn Hills Suites, Featherstone Road, Auburn Hills, MI, USA

End

Hyatt Place Detroit/Auburn Hills, North Opdyke Road, Auburn Hills, MI, USA

Submit

Route Segment: 1

2300 Featherstone Rd, Auburn Hills, MI 48326, USA to 4301 Orchard Lake Rd, West Bloomfield Township, MI 48323, USA

10.0 mi

Route Segment: 2

4301 Orchard Lake Rd, West Bloomfield Township, MI 48323, USA to 1967 Pine Ridge Ln, Bloomfield Hills, MI 48302, USA

About Us

Contact Us

FAQs

f

Copyright © 2018 Moove

Log In

Sign Up

Site Map

Moove  
Innovate the World

Sign Up
Log In

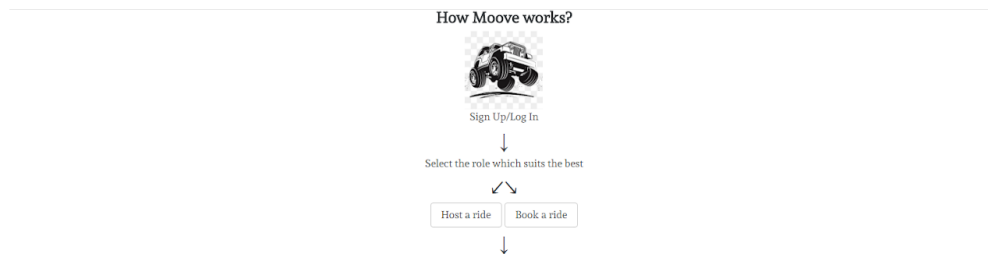
Register now for free &  
Share your ride

Email Address

By clicking Sign up, I agree to the Terms of Service and Privacy Policy.

Sign Up
OR
Log In

Moove is a ride-sharing application where the driver can host the ride and the passenger(s) can join the host and share the ride.



## 11. Testing

### 11.1. Unit Test Architecture and Strategy / Framework

- Our strategy is to ensure that any text problems, db issues, or other problem gets identified and squashed each and every week.

### 11.2. Unit test definition, test data selection

- Testing user data working for:
  - Email
  - Name
  - Date of Birth
  - Car type
  - Picture
  - Number of passengers

### 11.3. System Test Specification

- This is done using:
  - Unit testing
  - Acceptance testing
  - Integration testing

### 11.4. Test Reports per Spring

Sprint number	Test type	Variable tested	Result

1	Unit	Tested login unit	Working as intended
1	Acceptance	Tested login forms	Passwords form not yet working fully
2	Integration	Tested login screen with landing page	Working as intended
2	Acceptance	Tested contact us page working	Email sent to us not being sent properly, was fixed.
3	Unit	Testing ride/host functionality	Not working well
3	Acceptance	Tested host and find ride forms	Not initially saving to DB, eventually fixed in sprint 4.
4	Unit	Testing ride/host functionality	Can now function as intended
4	Acceptance	Tested host and find ride forms	Working as intended
5	Unit	Tested maps functionality	Could only figure out point A and B, not C or D if applicable
5	Acceptance	Tested google maps api auto filling locations in	Working as intended
6	Unit	Tested maps functionality	Working as intended
6	Integration	Tested if payment method is fully functioning with Stripe	Working as intended

## 12. Project Management

### 12.1. Project Plan

Sprint Number	Sprint Objectives	Estimated time
1	Begin initial documentation, map out application, login screen	15 hours each
2	Finish initial documentation, home screen, landing page, FAQ, contact page	20-25 hours each
3	Begin ride functionality, update documentation, make previous pages look better	25 hours each

4	Thank you page, profile page, payment page	15-20 hours each
5	Work on maps page, rework hosted/search ride, add stripe payment method	25 hours each
6	Fully implement maps, ride hosting, and payments. Fix all bugs	25 hours each
7	Bug fix, UI updates, perfect everything	20 hours each

## 12.2. Risk management

Risk	Potential Impact	Resolution
Maps functionality does not work as intended	No driver will be able to get from point A to B	Use google maps algorithm that is proven to work and implement that.
There are low driver counts	No One will be able to get a ride to their destination.	Offer driver incentives, that make it more appealing to be a host.

## 13. References

Google Maps Embedded API: <https://developers.google.com/maps/documentation/embed/>

Bootstrap Information: [https://www.w3schools.com/bootstrap/bootstrap\\_grid\\_system.asp](https://www.w3schools.com/bootstrap/bootstrap_grid_system.asp)

<https://www.affilorama.com/site-building/php>

Autocomplete Form:

<http://easyautocomplete.com/guide>

JQuery Library:

<https://cdnjs.com/libraries/jquery/1.11.2>

Accordion Format:

[https://www.w3schools.com/howto/howto\\_js\\_accordion.asp](https://www.w3schools.com/howto/howto_js_accordion.asp)

<https://getbootstrap.com/docs/4.0/components/collapse/>

Database tutorial to return the query based on user account:

[https://www.w3schools.com/php/php\\_mysql\\_select.asp](https://www.w3schools.com/php/php_mysql_select.asp)