Parking Management System

By Alec Agnese & Rami El Khatib

Group 18

COP 5339-001

Project Design Specification

Github Link: https://github.com/rkhat/ood-project

Youtube Link: https://www.youtube.com/watch?v=HjzBQH6cnNE

Project Title: Parking Management System

Functional Specifications

The parking management system is an application to manage the whole process of parking a customer's vehicle in a parking lot or garage. This includes parking the vehicle as well as paying for the spot. Only the customer will interact with the application. The project is going to be a simple desktop application and will run on any desktop OS (Windows, Linux or Mac).

When the application begins, it shows two choices; Member or Register. If the customer selects Register, they can signup for a new account. Clicking Member will allow old customers to log in to their account. Once logged in, The member can add a vehicle, remove a vehicle, select a vehicle to park, checkout, or access settings. Selecting a vehicle to park allows them to select a spot (if available). In the settings menu, they can add credits, change their password or log out.

Use Cases

1. Add Credits (Member)
2. Log out (Member)
3. Login (Member)
4. Sign Up
5. Park (Member)
6. Add Vehicle (Member)
7. Remove Vehicle (Member)
8. Checkout (Member)
9. Change password (Member)

Use Case 1: Add Credits

1. User carries out **Log In**
2. User clicks “Settings”
3. System displays Settings menu
4. User clicks “Add Credits”
5. System displays Credit Amount menu
6. System displays pop-up confirmation message
7. User clicks “Confirm”
8. System displays Settings menu

Use Case 2: Log out

1. User carries out **Log In**
2. User clicks “Settings”
3. System displays Settings menu
4. User clicks “Log out”
5. System displays Login Menu menu

Use Case 3: Log In

1. User clicks “Member”
2. System displays Login form
3. User enters username and password
4. System verifies login information
5. System displays Main Menu

Variation #1: Invalid login

1. In step 3, user enters invalid username or password
2. System display pop-up message “Invalid Login”
3. User clicks “Ok”

4. Continue with step 2

Use Case 4: Sign Up

1. User clicks “Sign up”
2. System displays Sign-up Form
3. User enters username and password
4. System verifies username unique
5. System display pop-up message “Account Created Successfully”
6. User clicks “Ok”
7. System logs member in and displays Main Menu

Variation #1: Username taken

1. In step 3, user enters unavailable username
2. System displays pop-up message “Username taken”
3. User clicks “Ok”
4. Continue with step 2

Use Case 5: Park

1. User carries out **Log In**
2. User selects vehicle from list
3. System displays parking spot map
4. User selects parking spot
5. System displays pop-up confirmation message
6. User clicks “Confirm”
7. System displays Main Menu

Variation #1: No Spots Available

1. User carries out steps 1-2
2. System displays pop-up message “No Spots Available”
3. User clicks “Ok”
4. System displays Main Menu

Use Case 6: Add vehicle

1. User carries out **Log In**
2. User enters plate number and clicks “add”
3. System adds vehicle to list

Use Case 7: Remove Vehicle

1. User carries out **Log In**

2. User selects existing vehicle to remove
3. System removes the vehicle from list

Use Case 8: Check Out

1. User carries out **Log In**
2. User clicks “Checkout”
3. System verifies that member has credits
4. System display pop-up confirmation message
5. User clicks “Confirm”
6. System displays Main Menu

Variation #1: No credits

1. User carries out steps 1-2
2. System displays pop-up message “Insufficient credits”
3. User clicks “cancel”
4. System displays Main Menu

Use Case 9: Change password

1. User carries out **Log In**
2. User clicks “Settings”
3. System displays Settings menu
4. User clicks “Change Password”
5. System displays Change Password form
6. User enters old password once and new password twice
7. System changes password and displays Main Menu

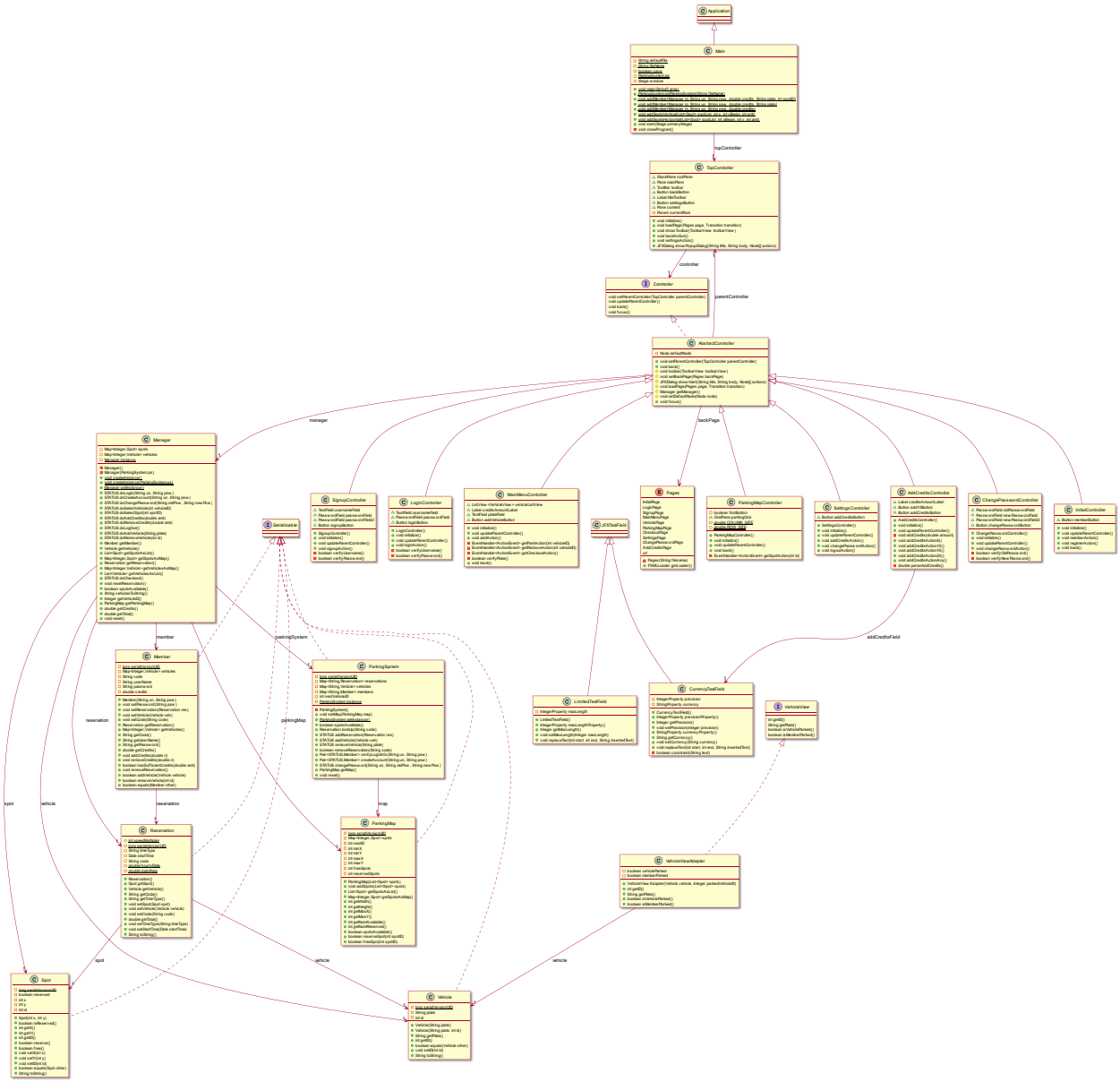
Variation #1: Incorrect old password

1. In step 6, user enters invalid old password
2. System displays pop-up message “Invalid old password”
3. User clicks “Ok”
4. Continue with step 5

Variation #2: New password not matching

1. In step 6, user enters valid old password but non-matching new passwords
2. System displays pop-up message “New passwords don’t match”
3. User clicks “Ok”
4. Continue with step 5

Class Diagram



CRC Cards

Abstract **Manager**	
• Load current logged in Member's data • Carry out user actions • Create reservations	• Member • Reservation • ParkingSystem • ParkingMap • UI

Reservation	
• Manage reservation details	

ParkingMap	
• Manage parking spots	• Spot

Spot	
• Manage spot details	

ParkingSystem	
• Manage reservations • Manage accounts • Manage parking map	• Member • Reservation • ParkingMap

UI	
• Take user input • Relay user input to controller	• Manager

Member	
• Manage username and password • Store member vehicles • Store member reservation • Manage member credits	

Vehicle	
• Manage vehicle details	

TopController	
• Manages swapping pages • Manages the toolbar • Shows popup dialogs	• Controller

Interface Controller AbstractController	
• Back action • Which node to focus when swapping pages	

Abstract	**AbstractController**	Controller
• Popup dialog on each page. • Loading a page • Getting the manager • Settings the default node to focus		• Manager

	GUIController	AbstractController
• Manages its corresponding page		

	CurrencyTextField	JFXTextField
• Displays currency number		

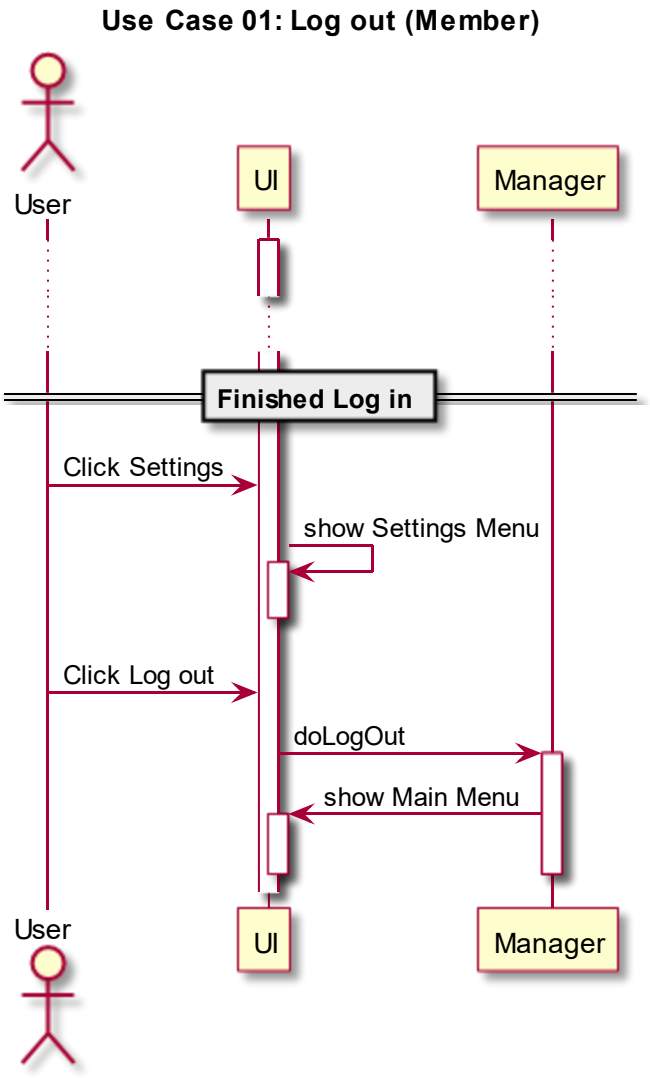
	LimitedTextField	TextField
• Limit number of characters		

	ToolbarView	
• contains data about how the toolbar should be viewed		

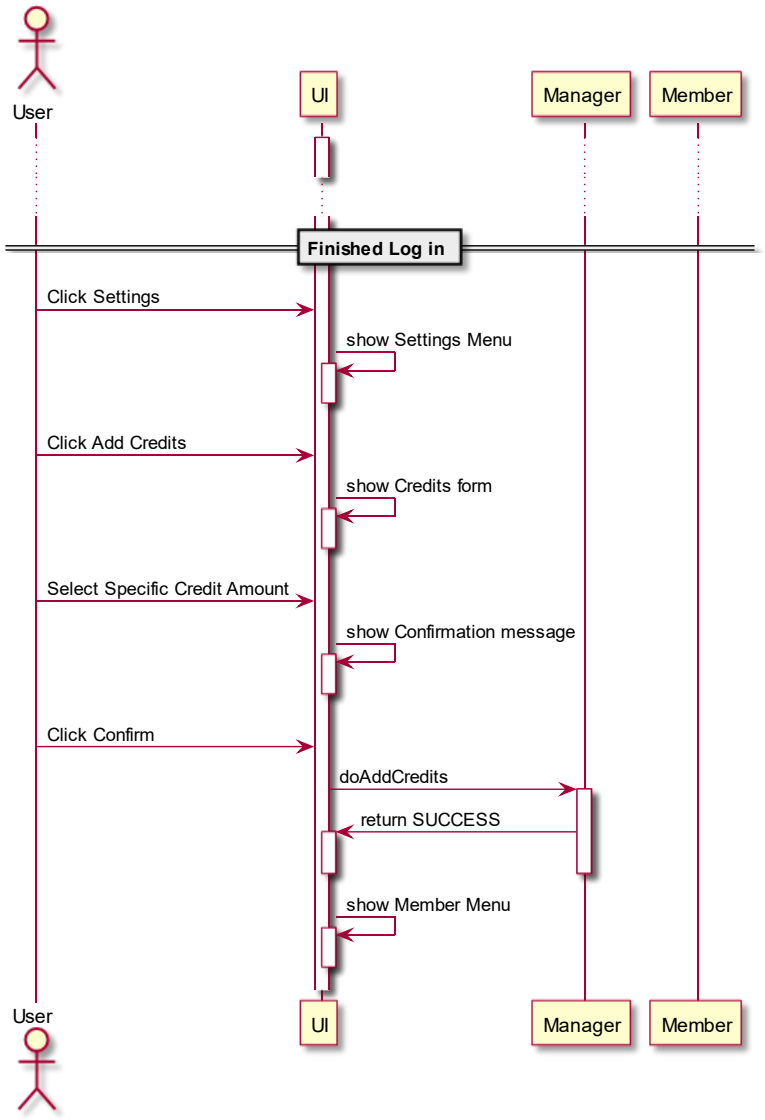
Interface	**VehicleView**	
• What to display for the vehicle in the listview		

VehicleViewAdapter	VehicleView
• Converts from Vehicle to VehicleView	

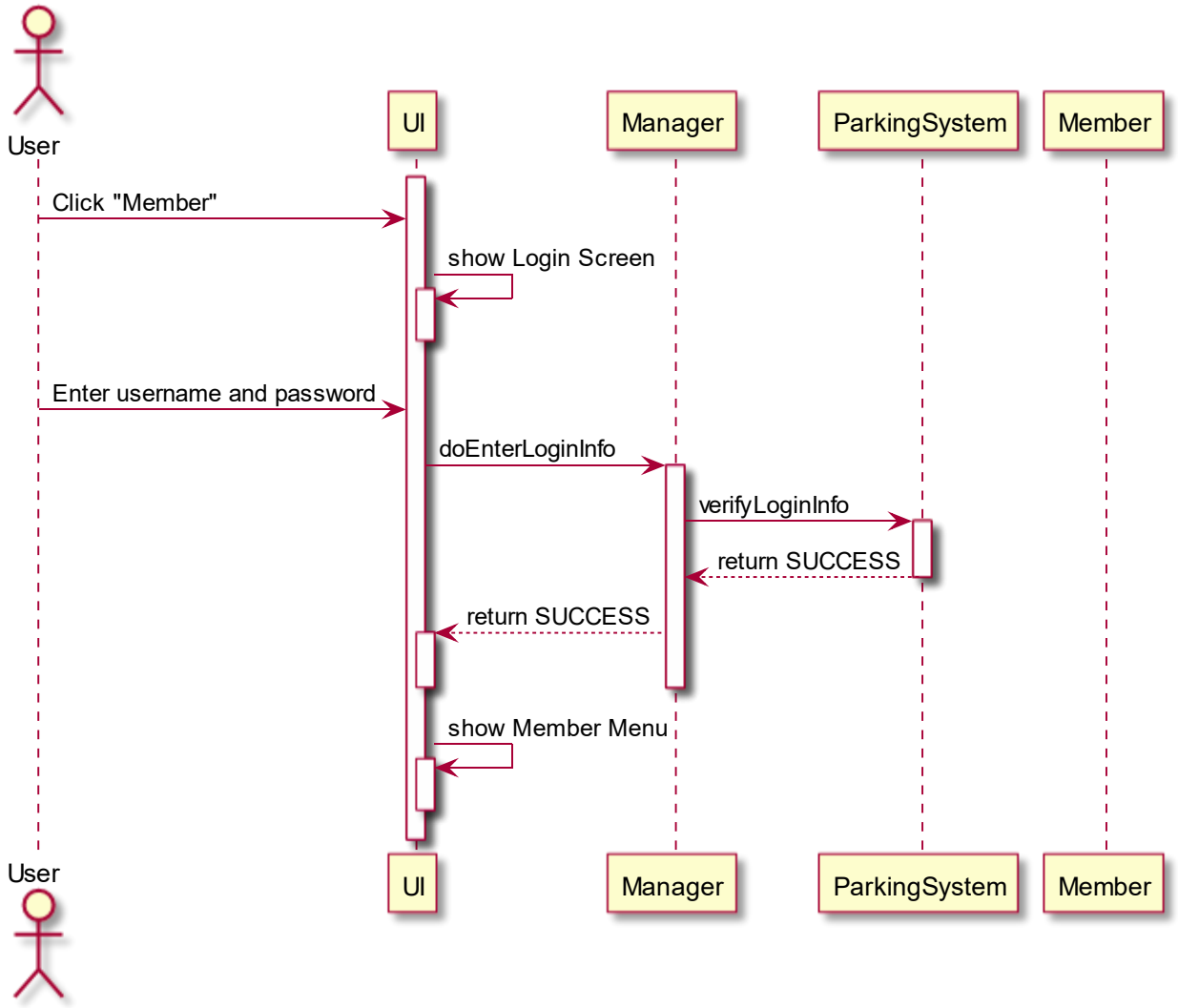
Sequence Diagrams



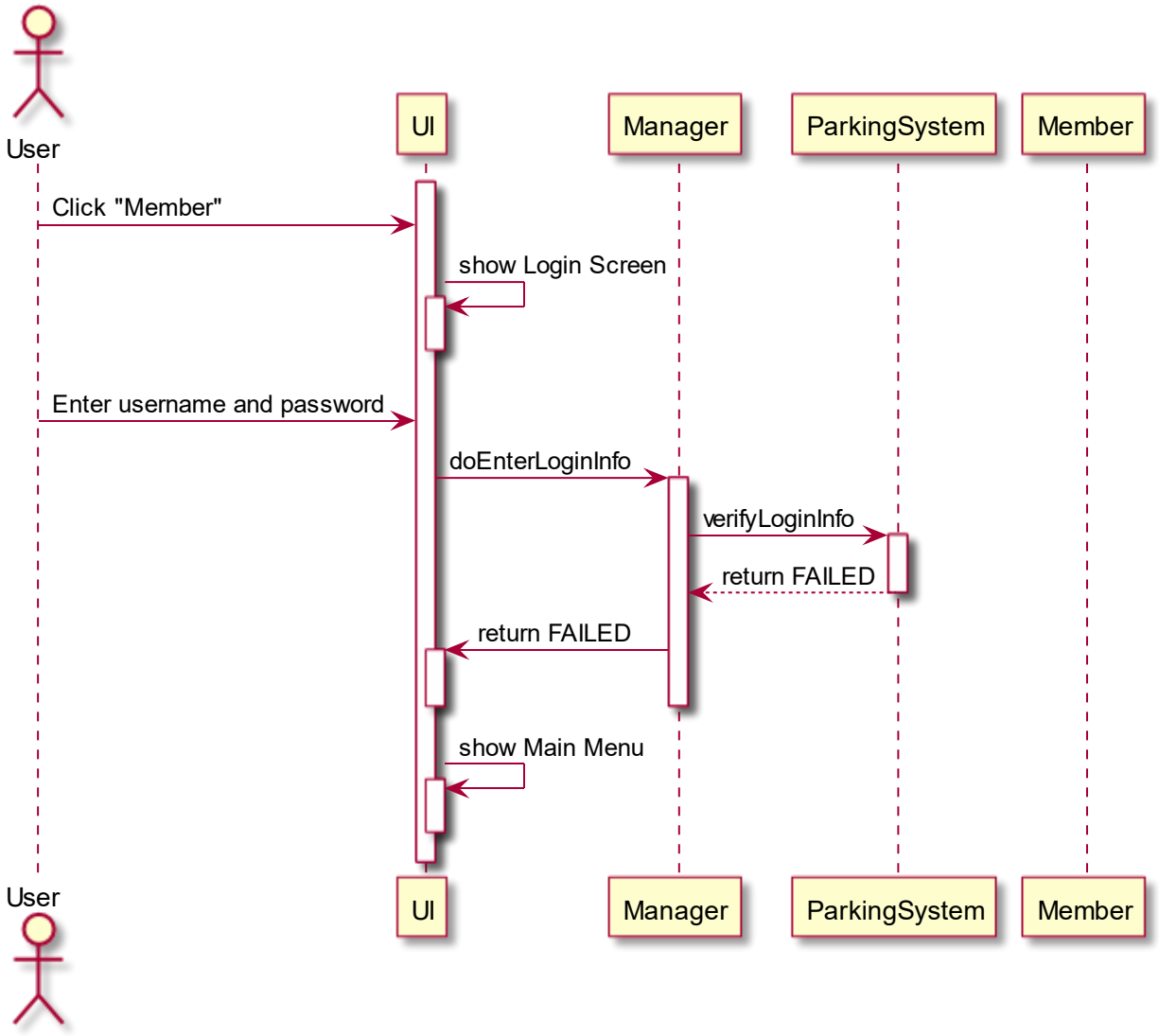
Use Case 02: Add Credits (Member)



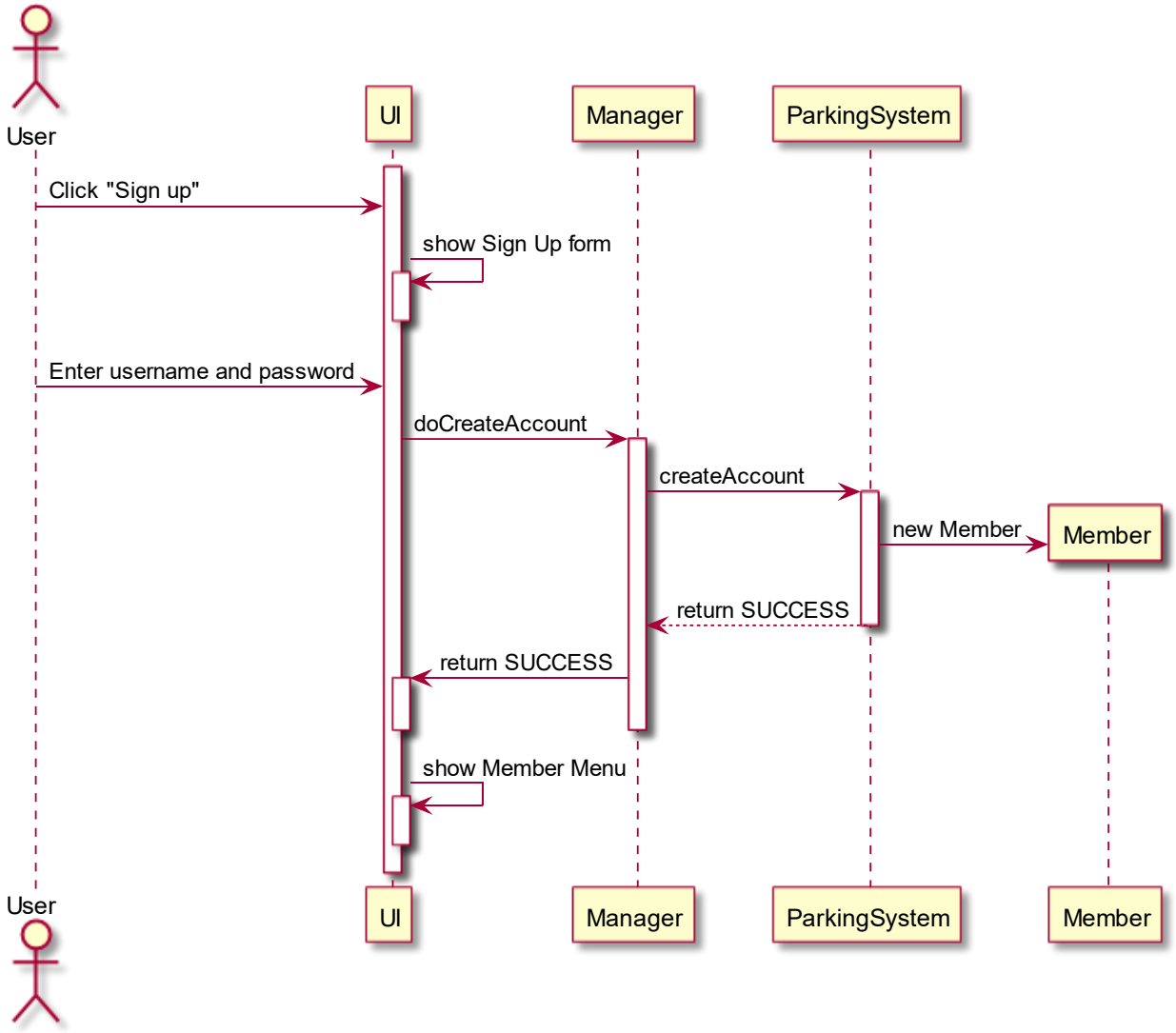
Use Case 3: Log In (Member)



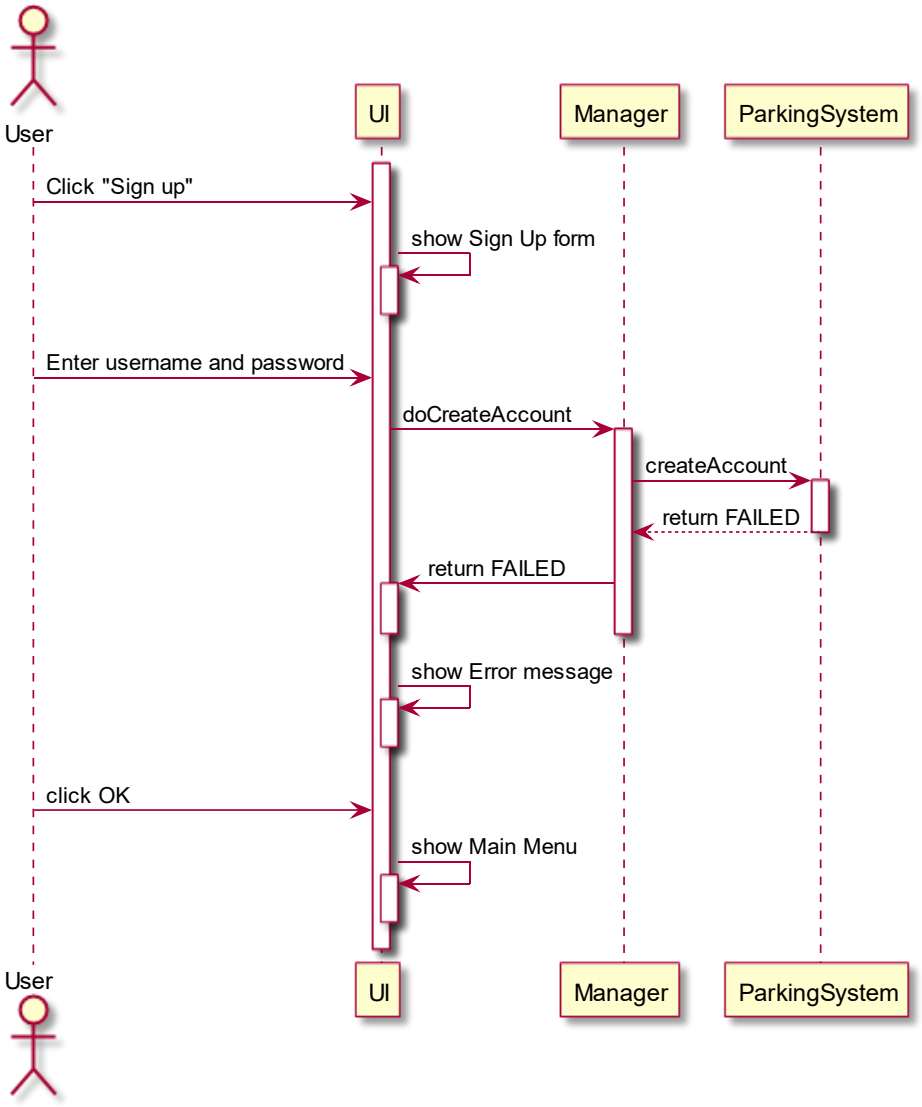
Variation #1: Invalid login



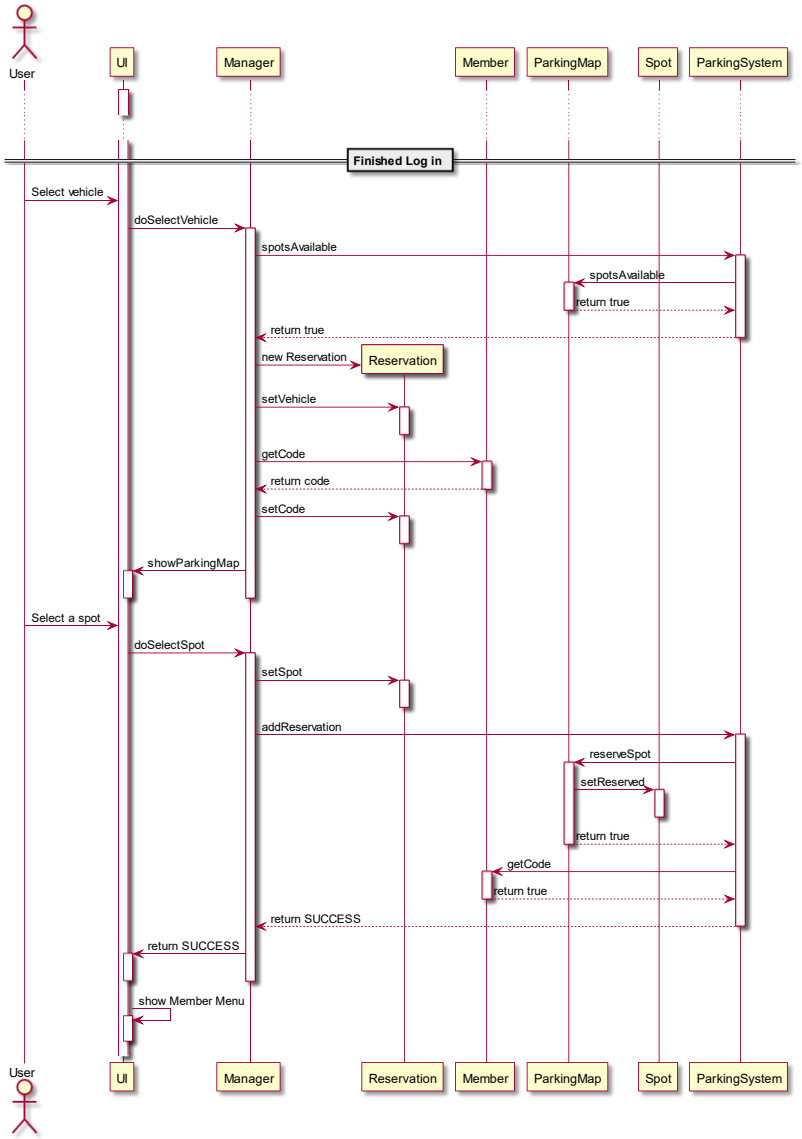
Use Case 4: Sign Up



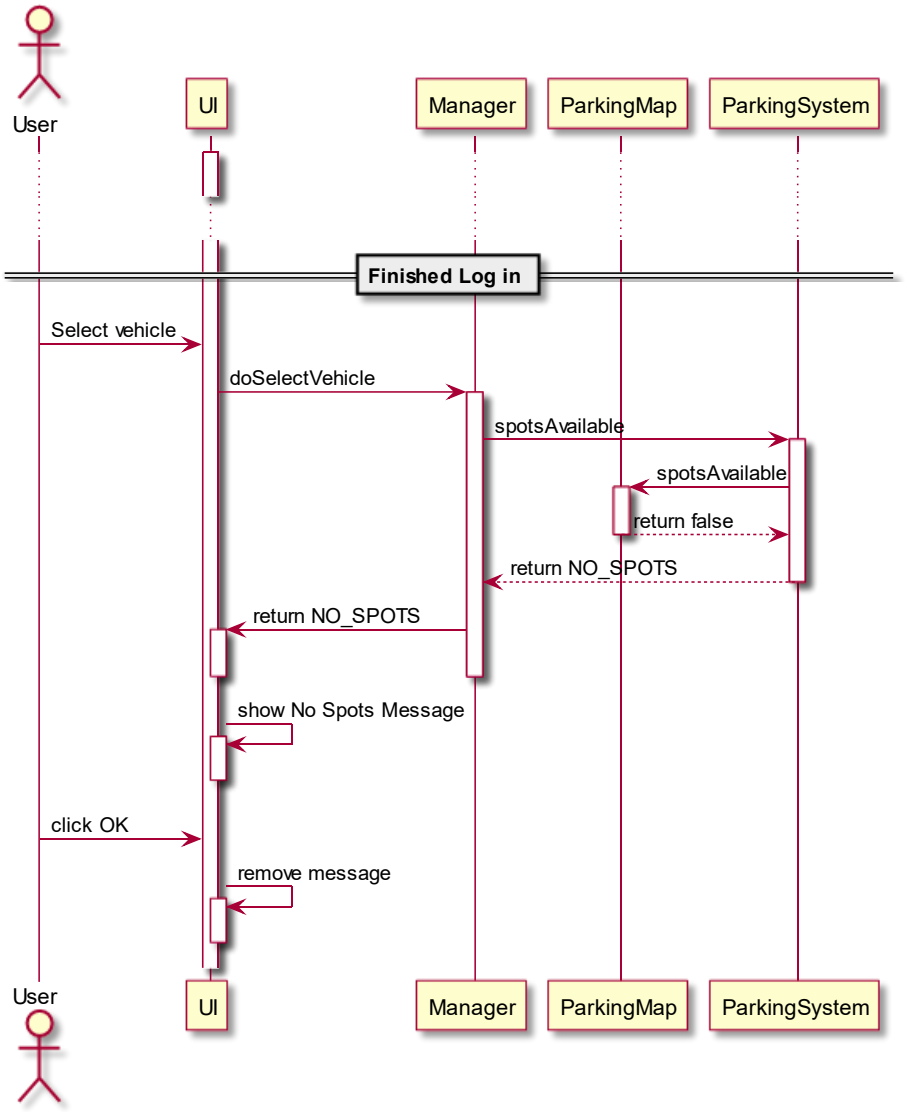
variation #1: username taken



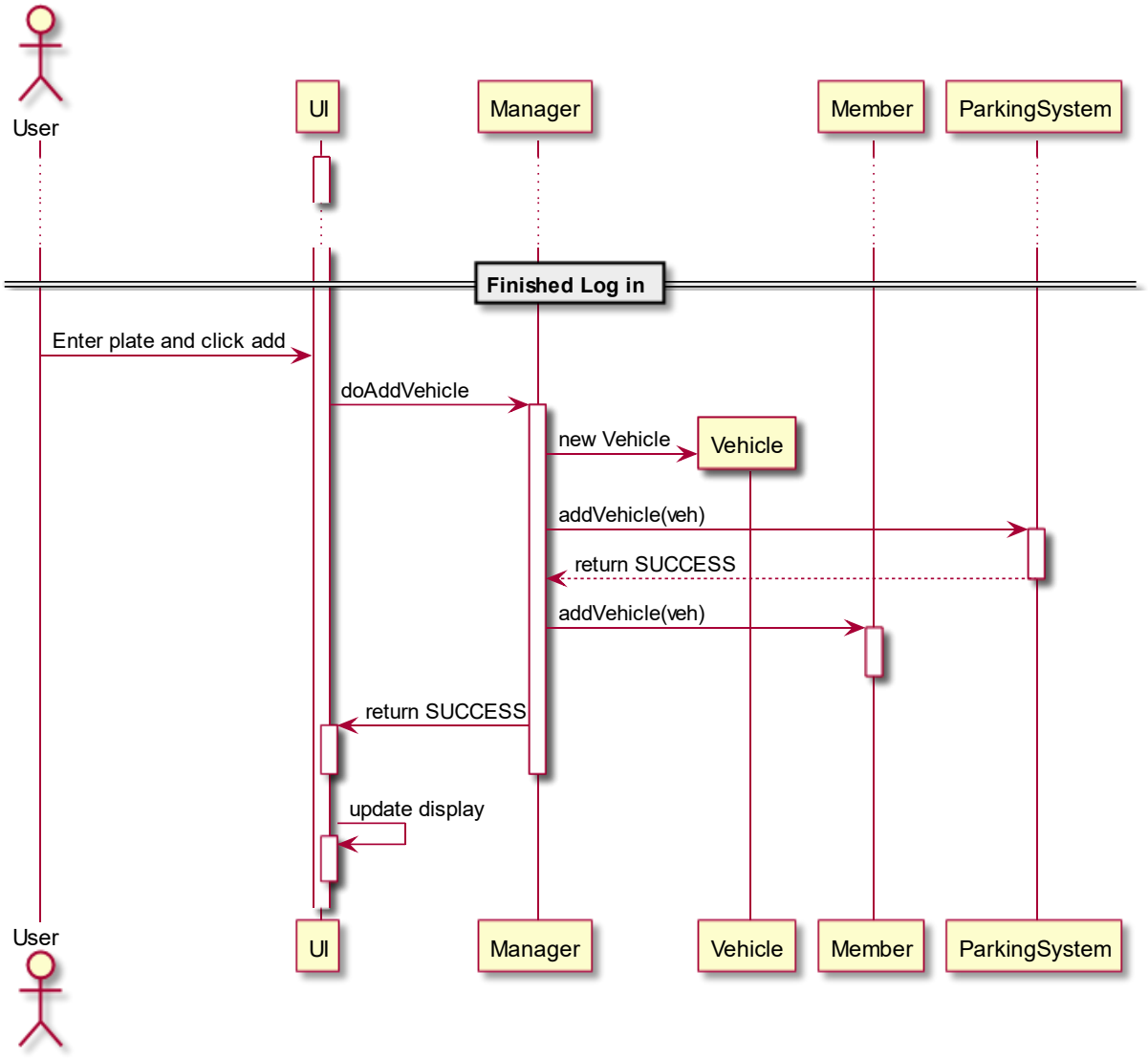
Use Case 5: Park (Member)



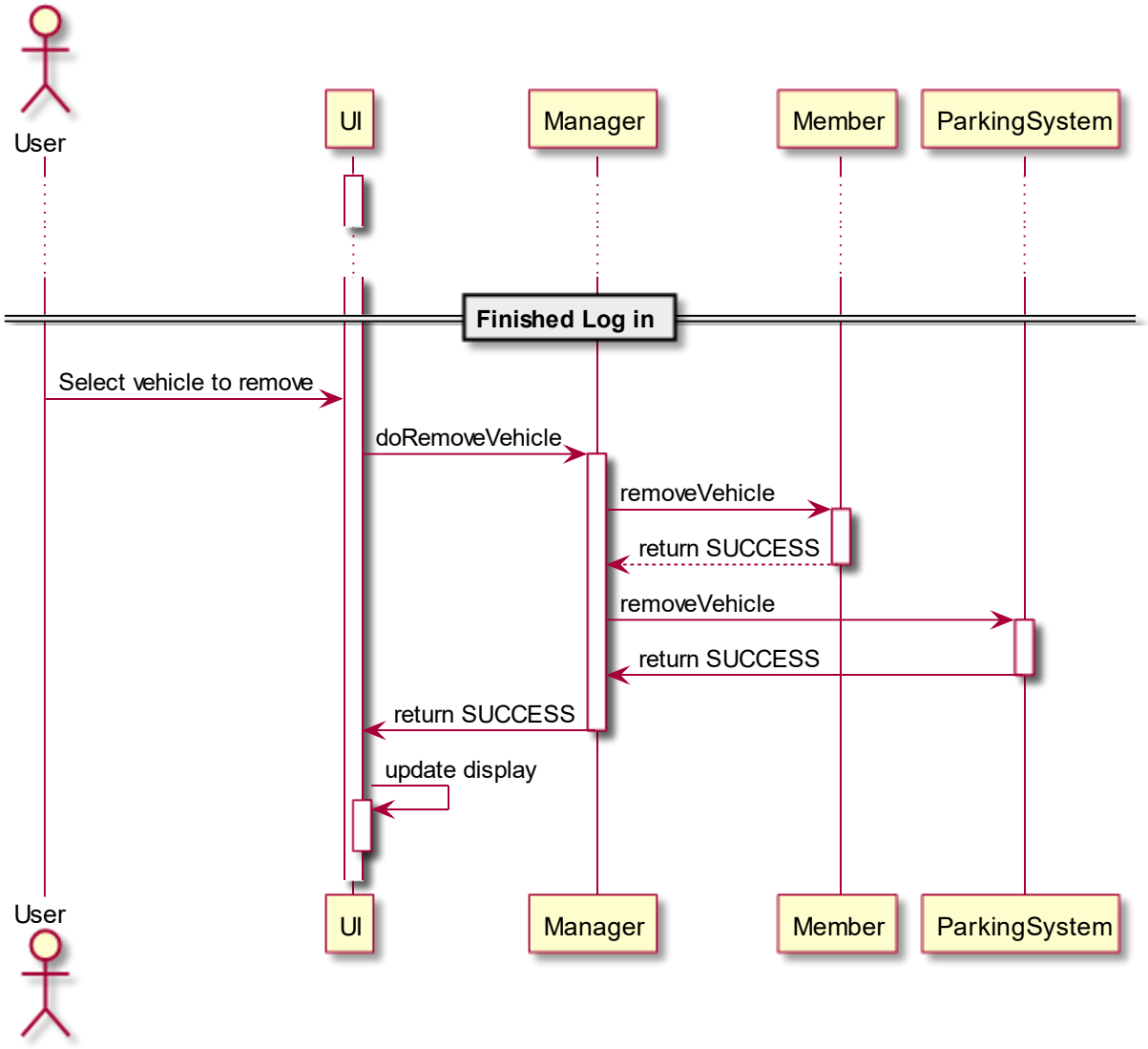
Variation #1: No Spots Available (Member)



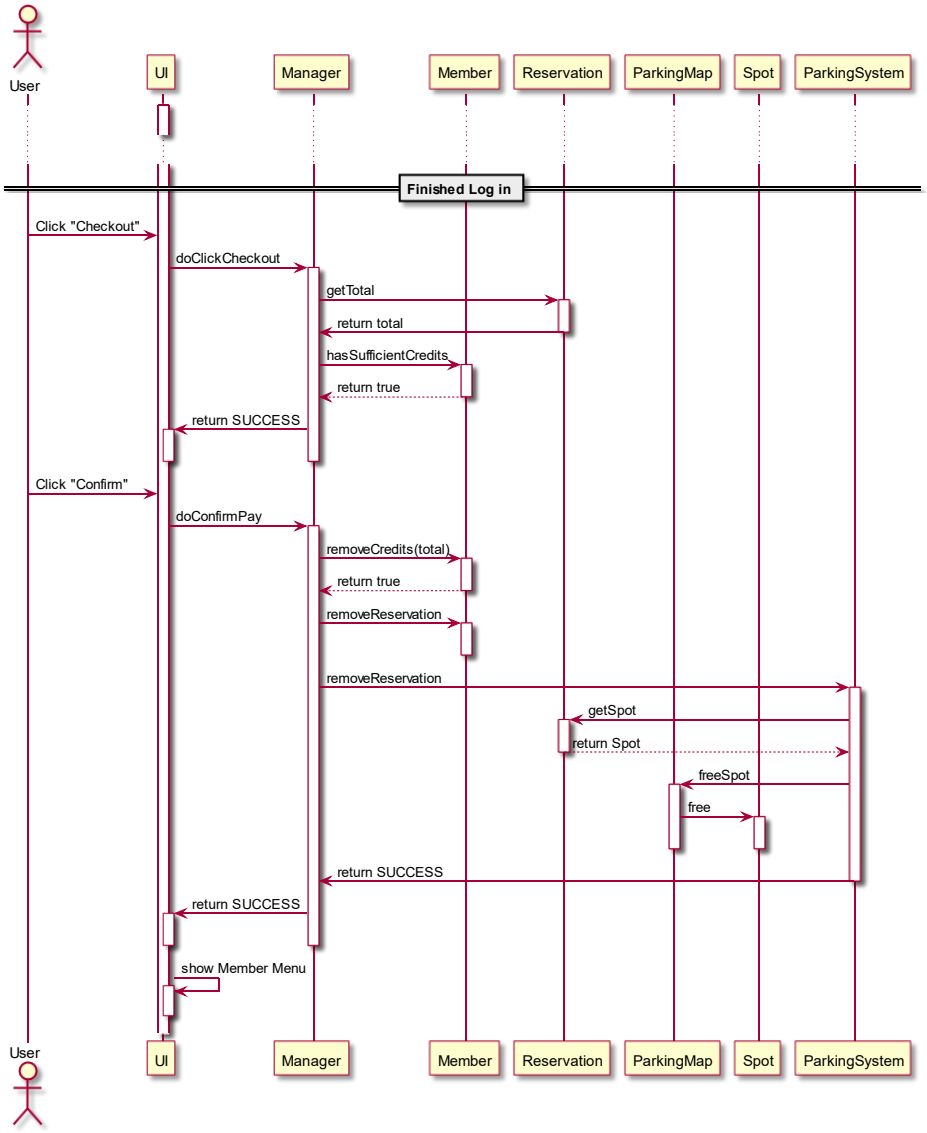
Use Case 6: Add Vehicle



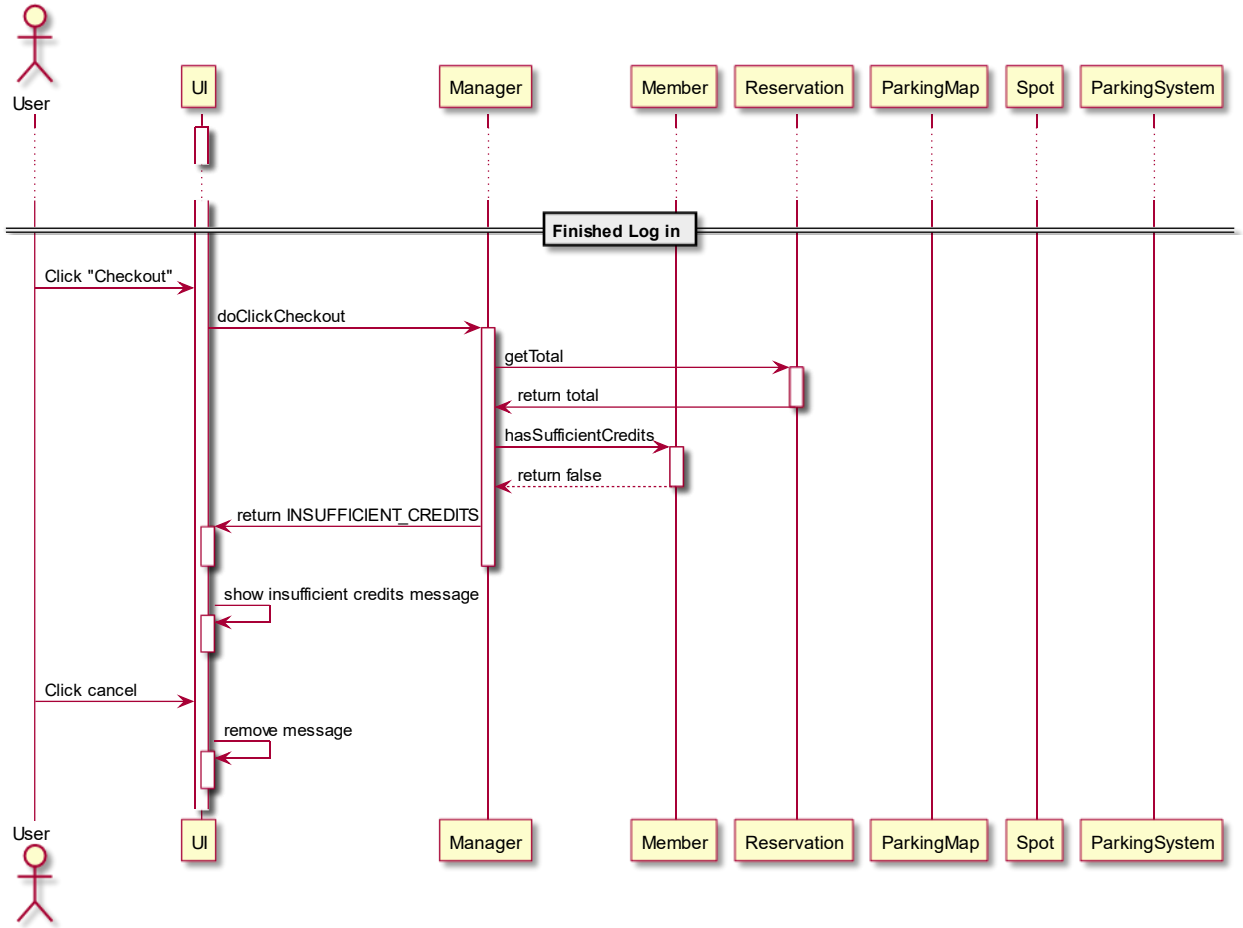
Use Case 7: Remove Vehicle



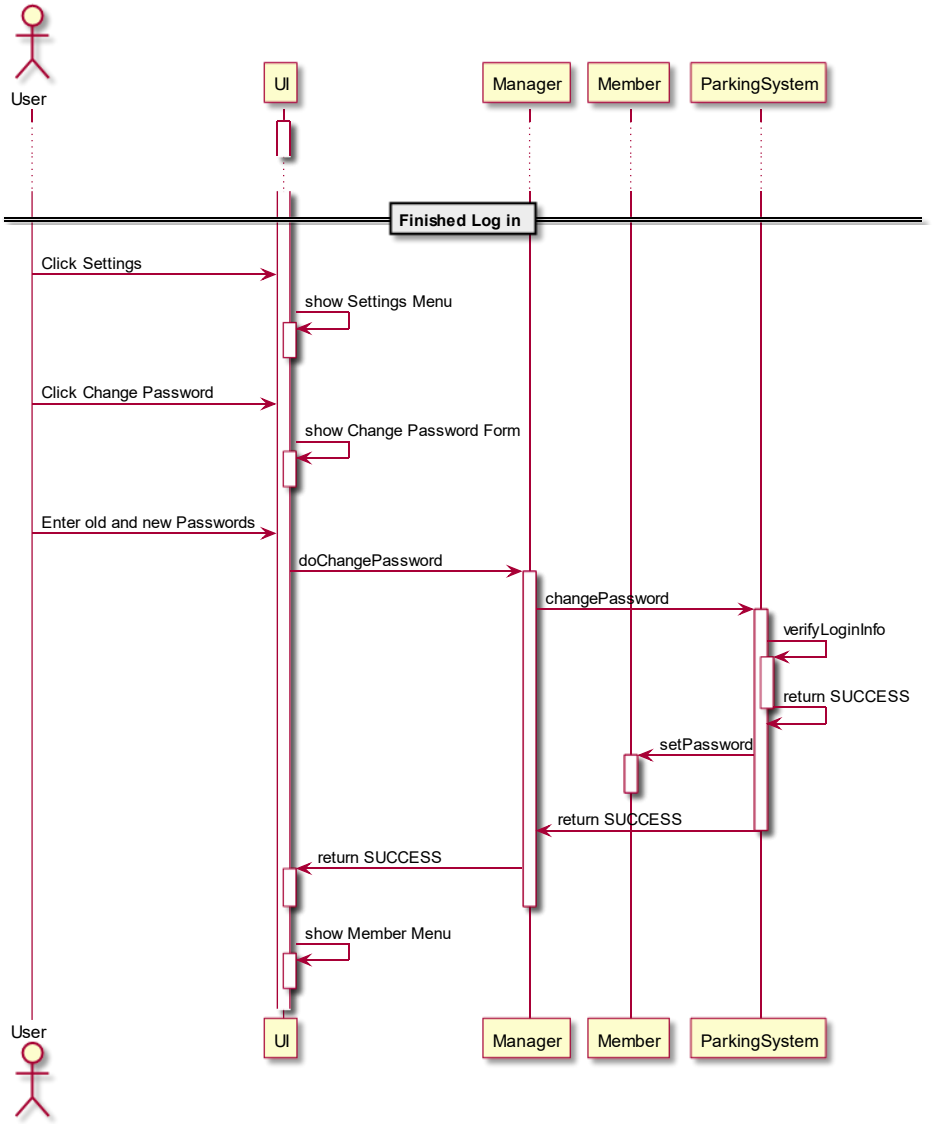
Use Case 8: Check out (Member)



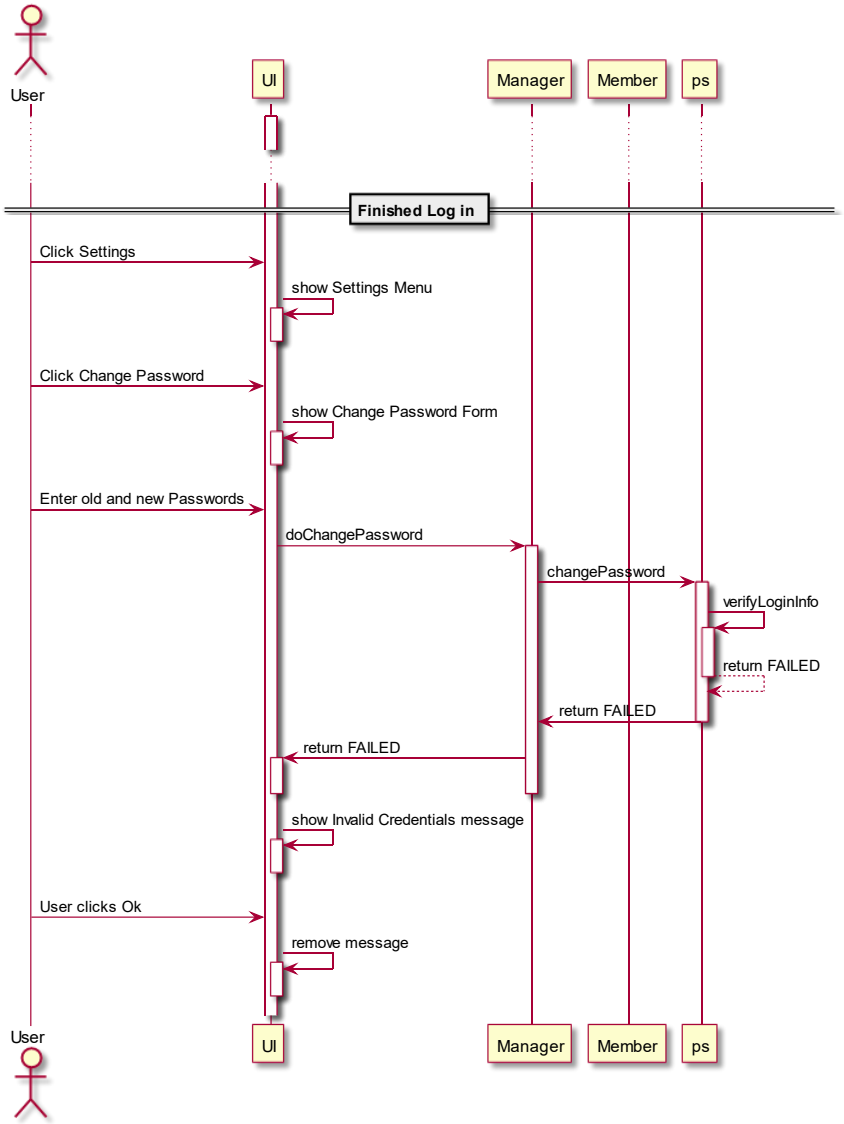
Variation #1: Insufficient credits - Return



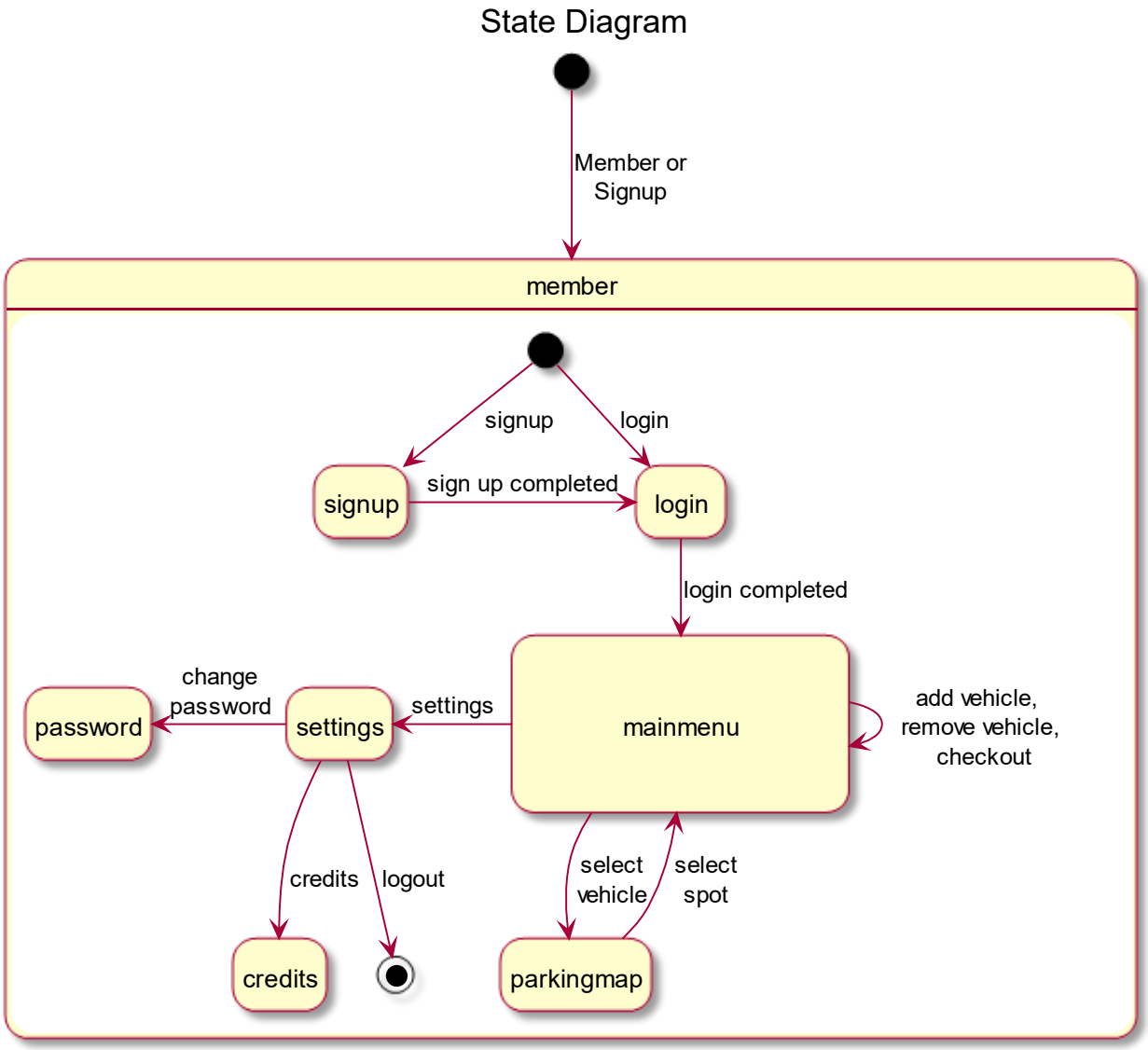
Use Case 9: Change password (Member)



Variation #1: Incorrect old password

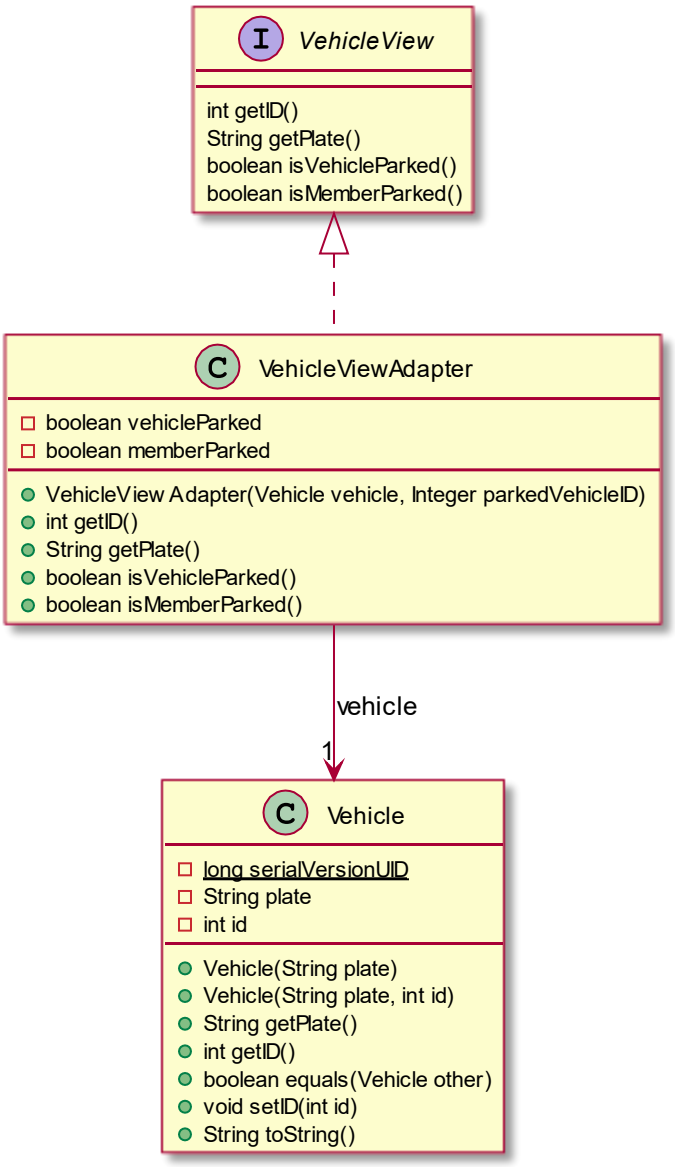


State Diagram

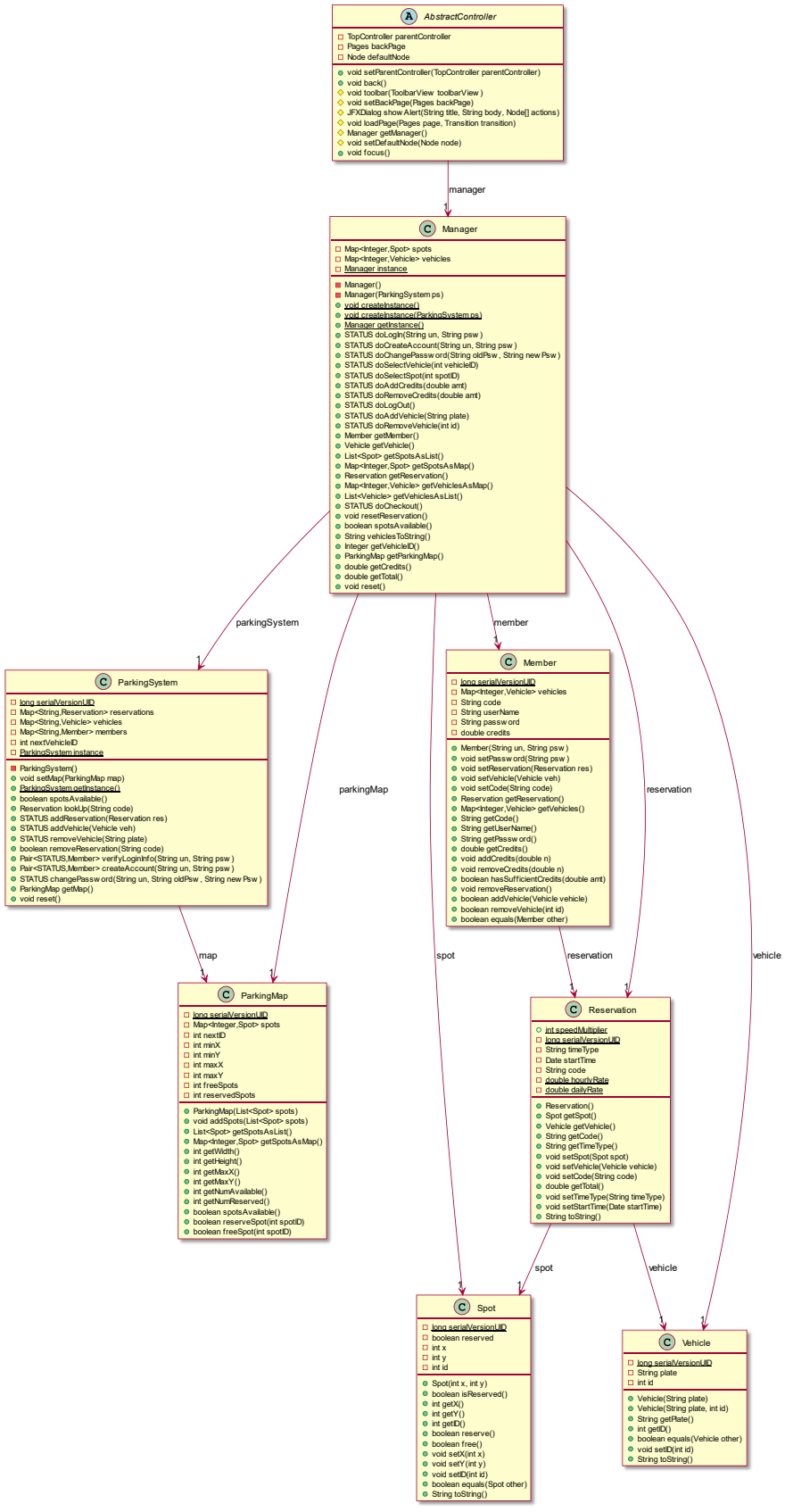


Design Patterns

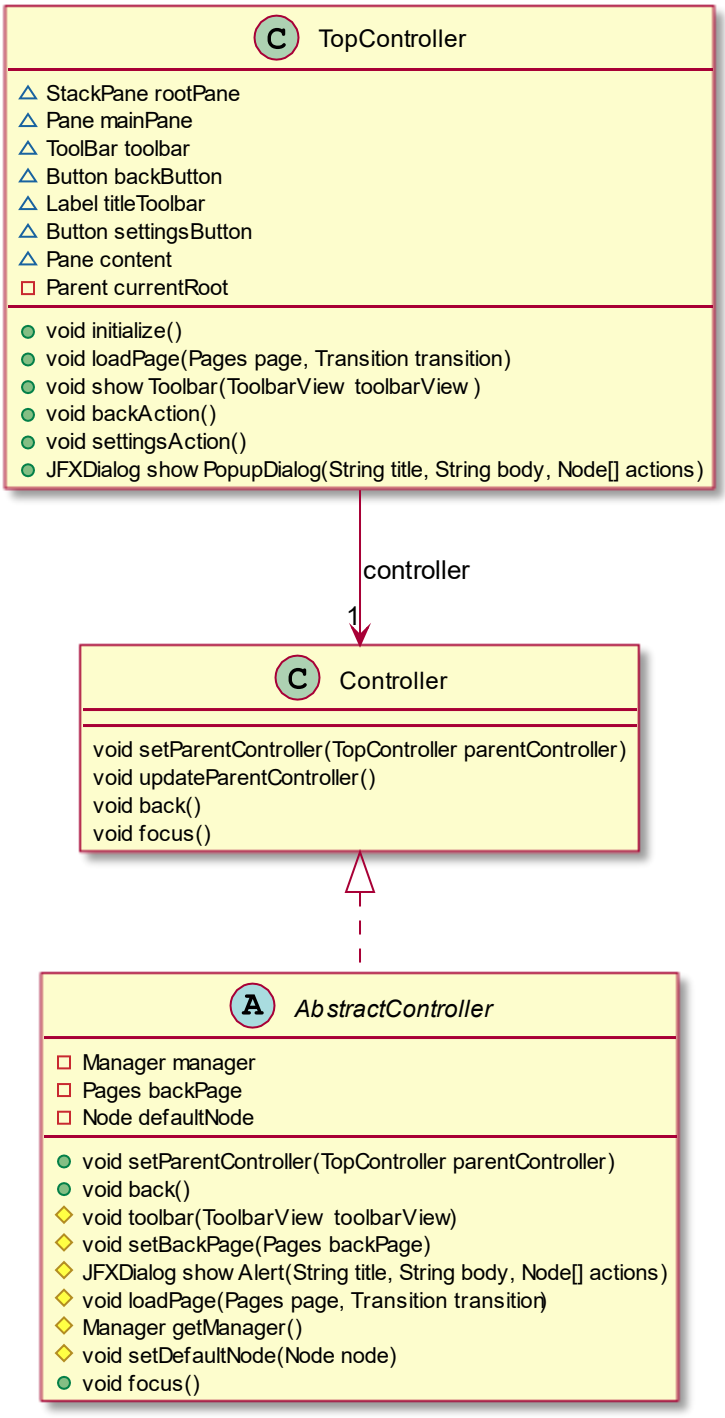
Adapter Pattern



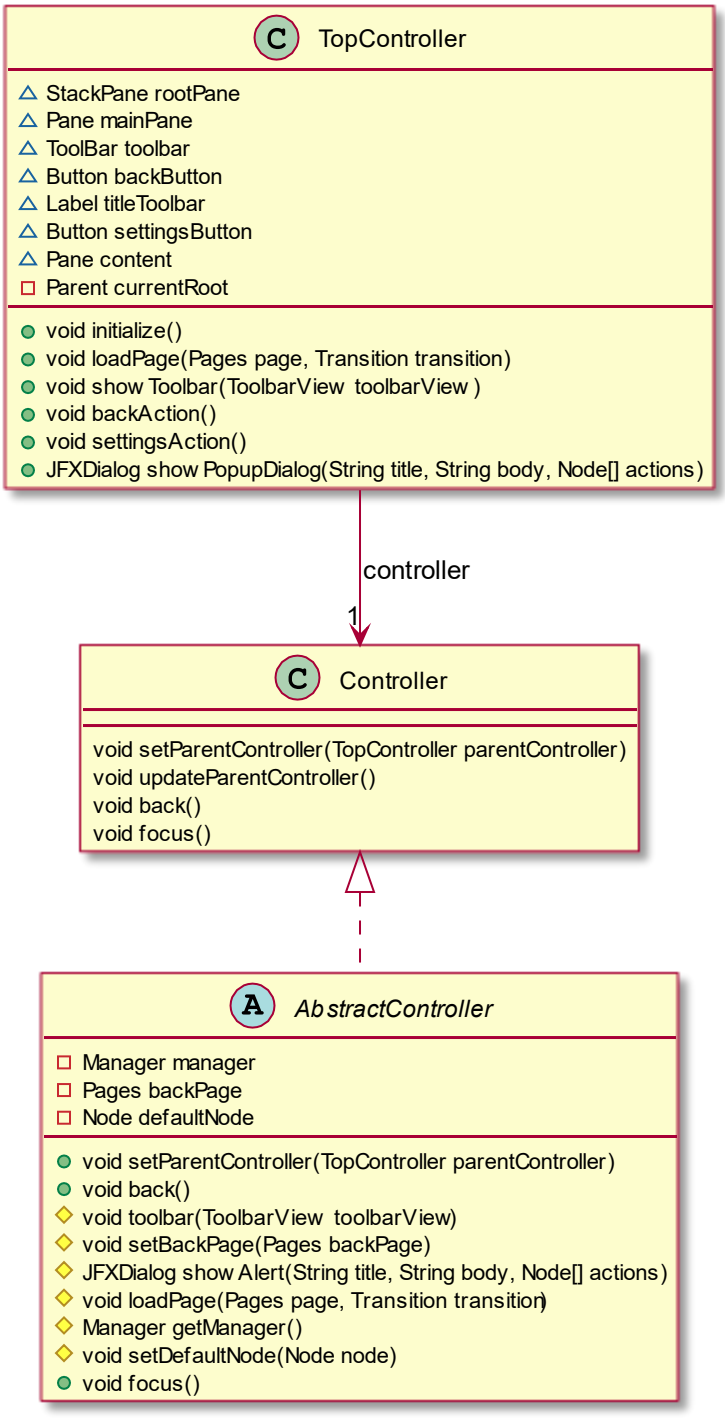
Facade Pattern



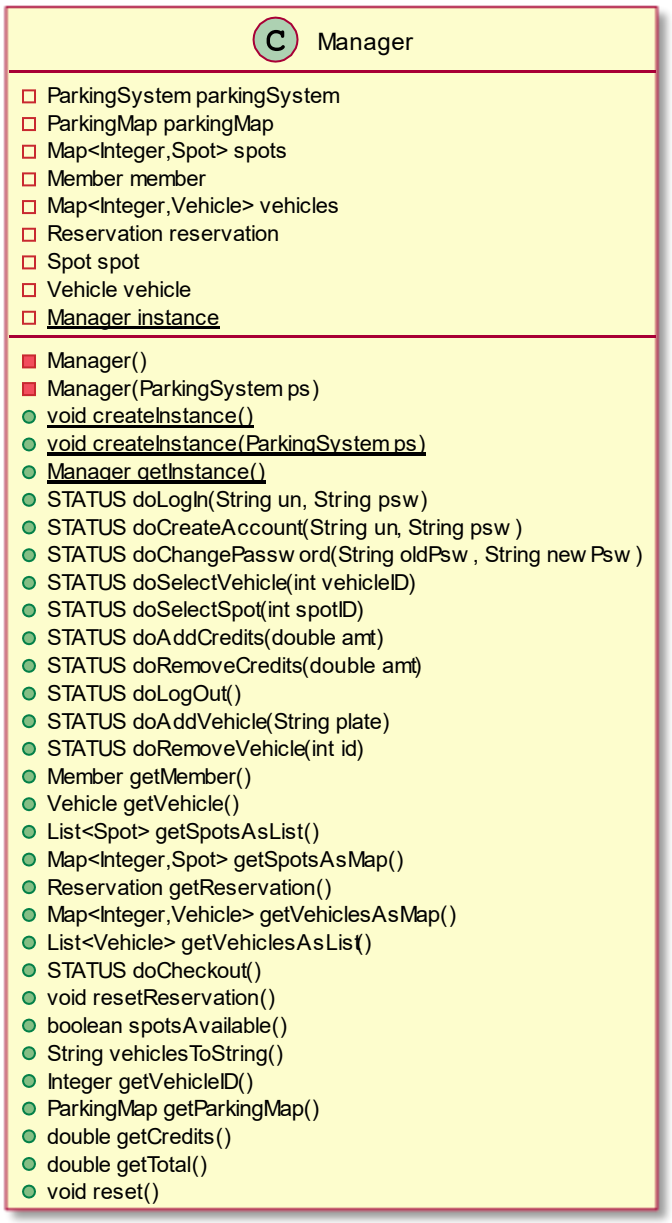
Observer Pattern



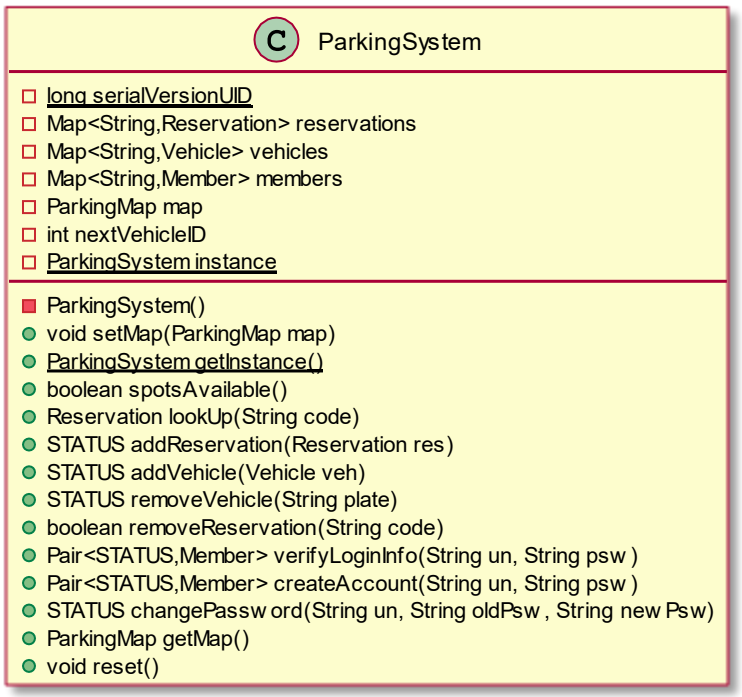
Strategy Pattern



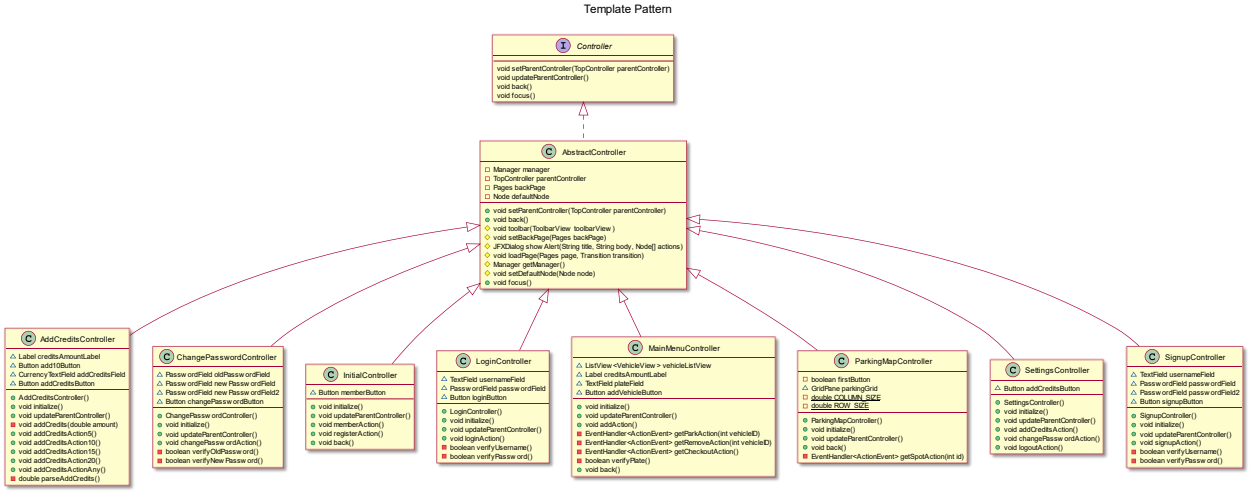
Singleton Pattern #1



Singleton Pattern #2



Template Pattern



What We Learned

Working on our project this semester provided us with a lot of great hands-on experience. Collaborating on a project with multiple developers can be a challenge, but with the proper preparation and communication, it becomes very manageable. The two most valuable lessons we learned in terms of collaboration were the use of Git and efficient division of the work load. Using Git greatly facilitated collaboration by allowing us to work separately while having the ability to easily share changes with a simple “push” or “pull”. Additionally, by dividing the work in such a way that neither of us was ever working on the same file simultaneously, we mitigated the risk of conflicting changes which can quickly cause complications.

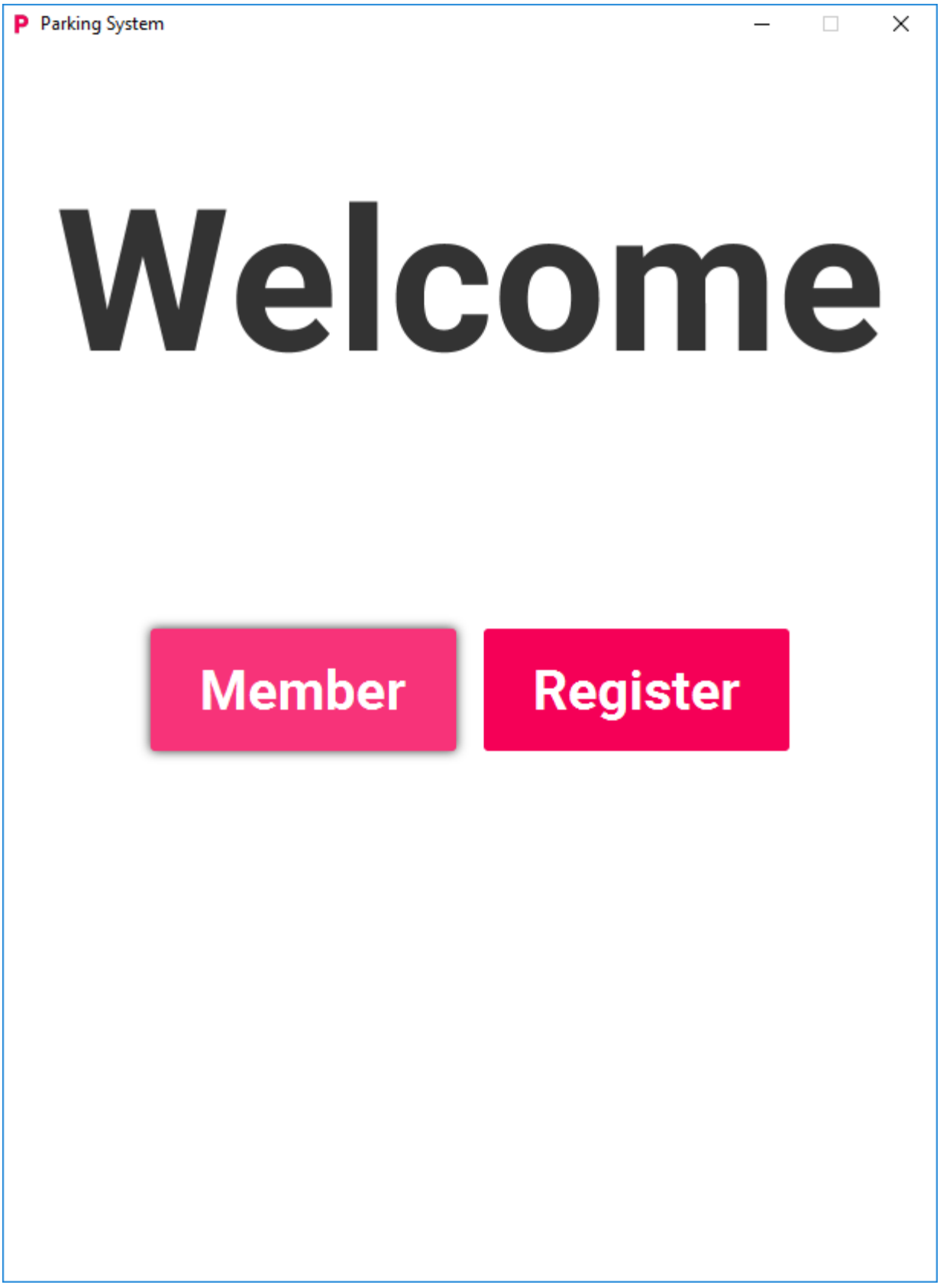
On the Object Oriented Design end, the main lesson we learned was the meaningful use of Design Patterns. The design patterns we used were Singleton, Façade, Adapter, Observer, Template, and Strategy, which were discussed in the Design Patterns section. By using them, we saved a lot of time that would have otherwise been spent “reinventing the wheel”.

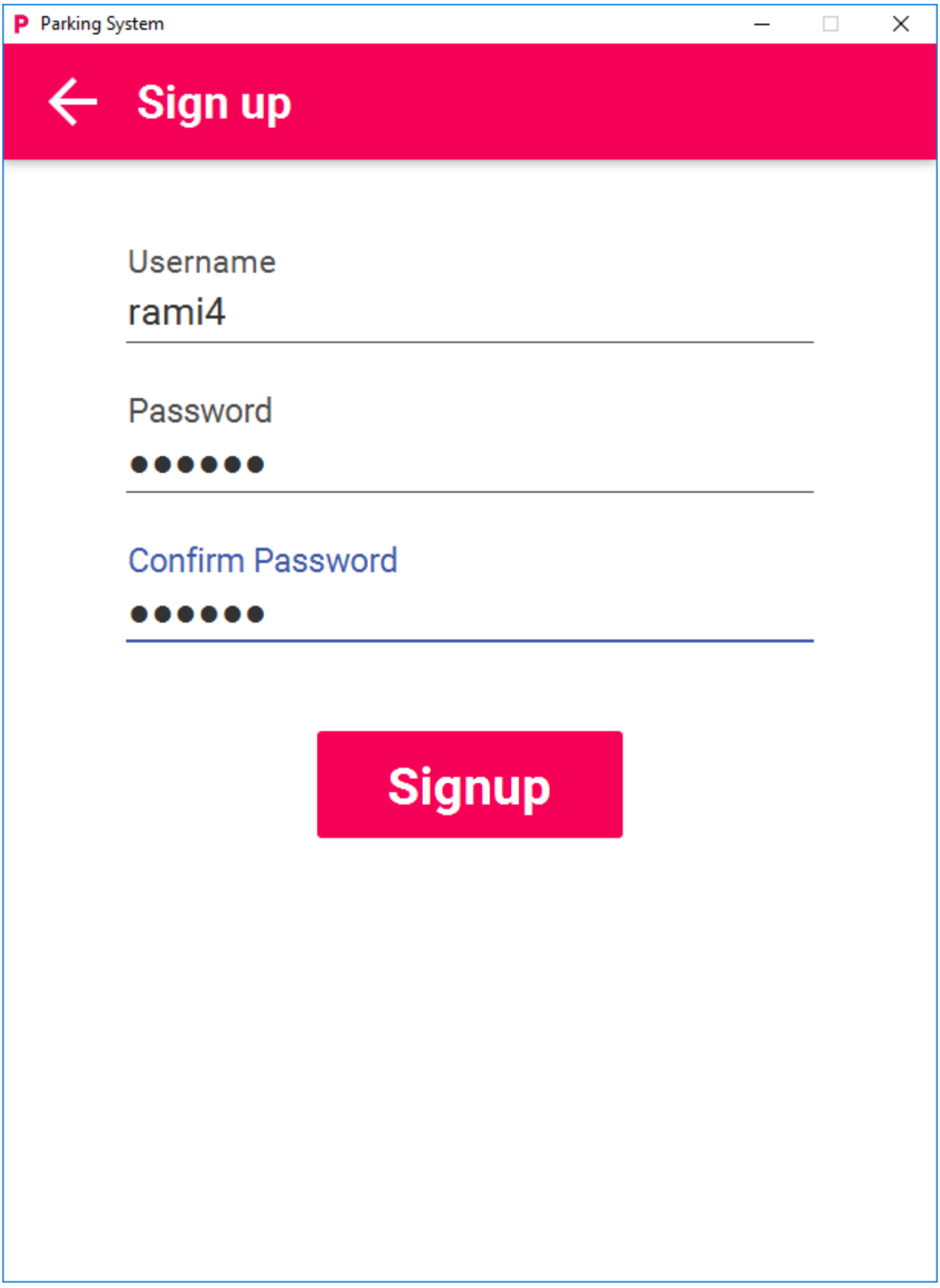
Finally, we learned how to create a GUI application using MVC to tie together the model, view, and controller. Doing this correctly made it much easier to make changes to one aspect of the program without requiring changes elsewhere.

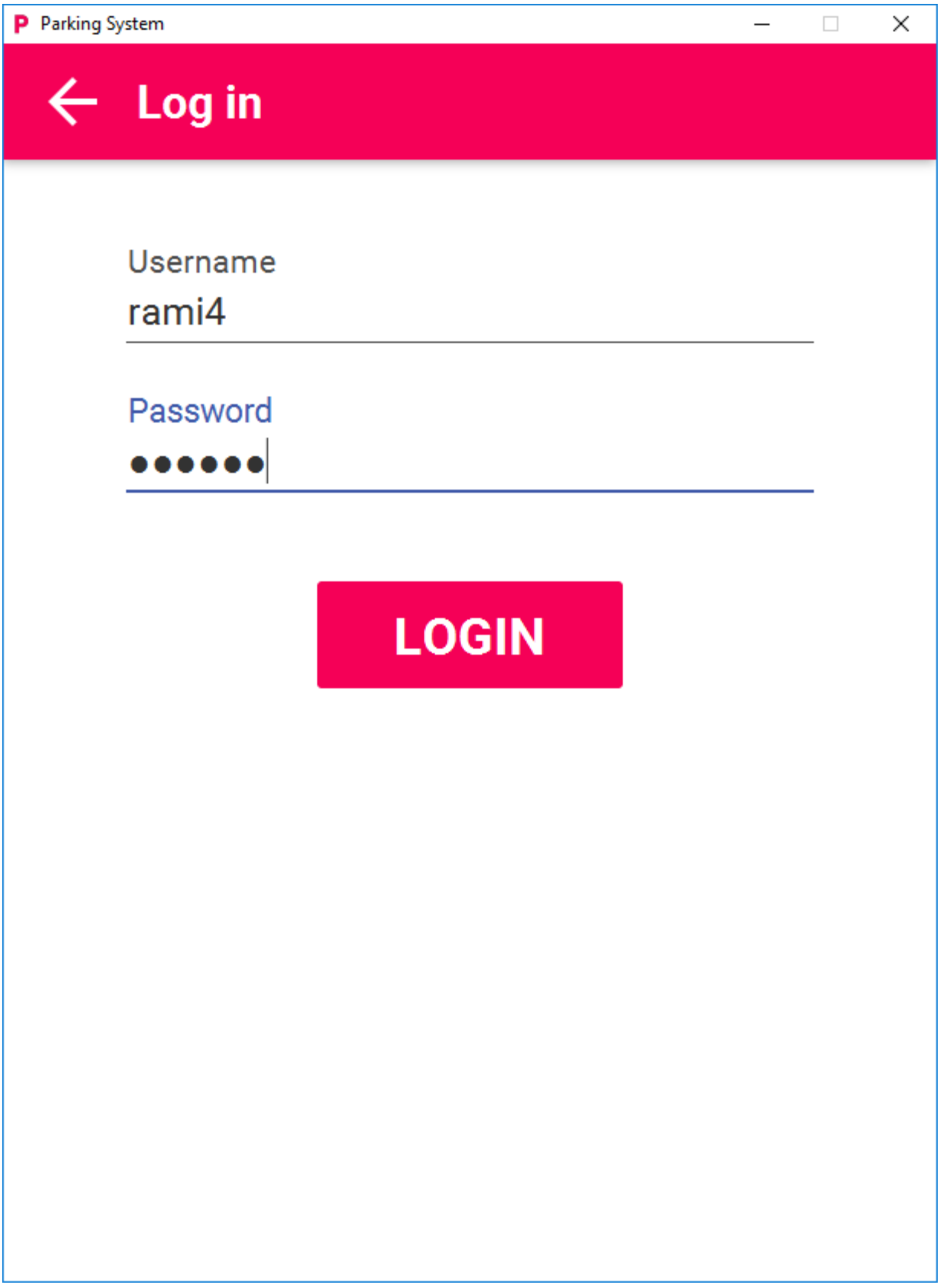
Conclusion

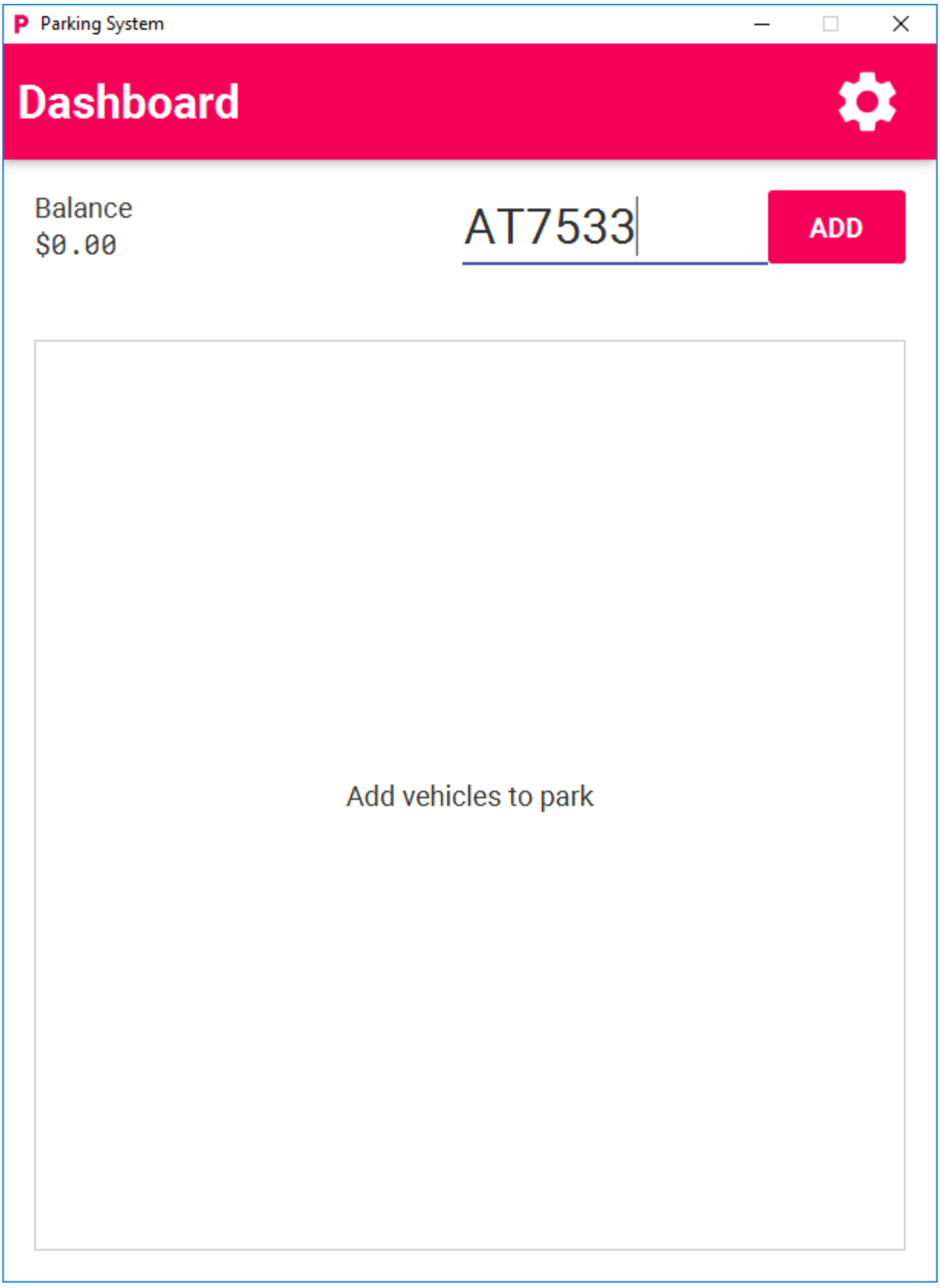
The project satisfied all the goals required for this project. A number of unit tests The design patterns used for the project were Singleton, Façade, Adapter, Observer, Template, and Strategy pattern. Some future work can be done in this project. Below is some of the ideas that can be done.

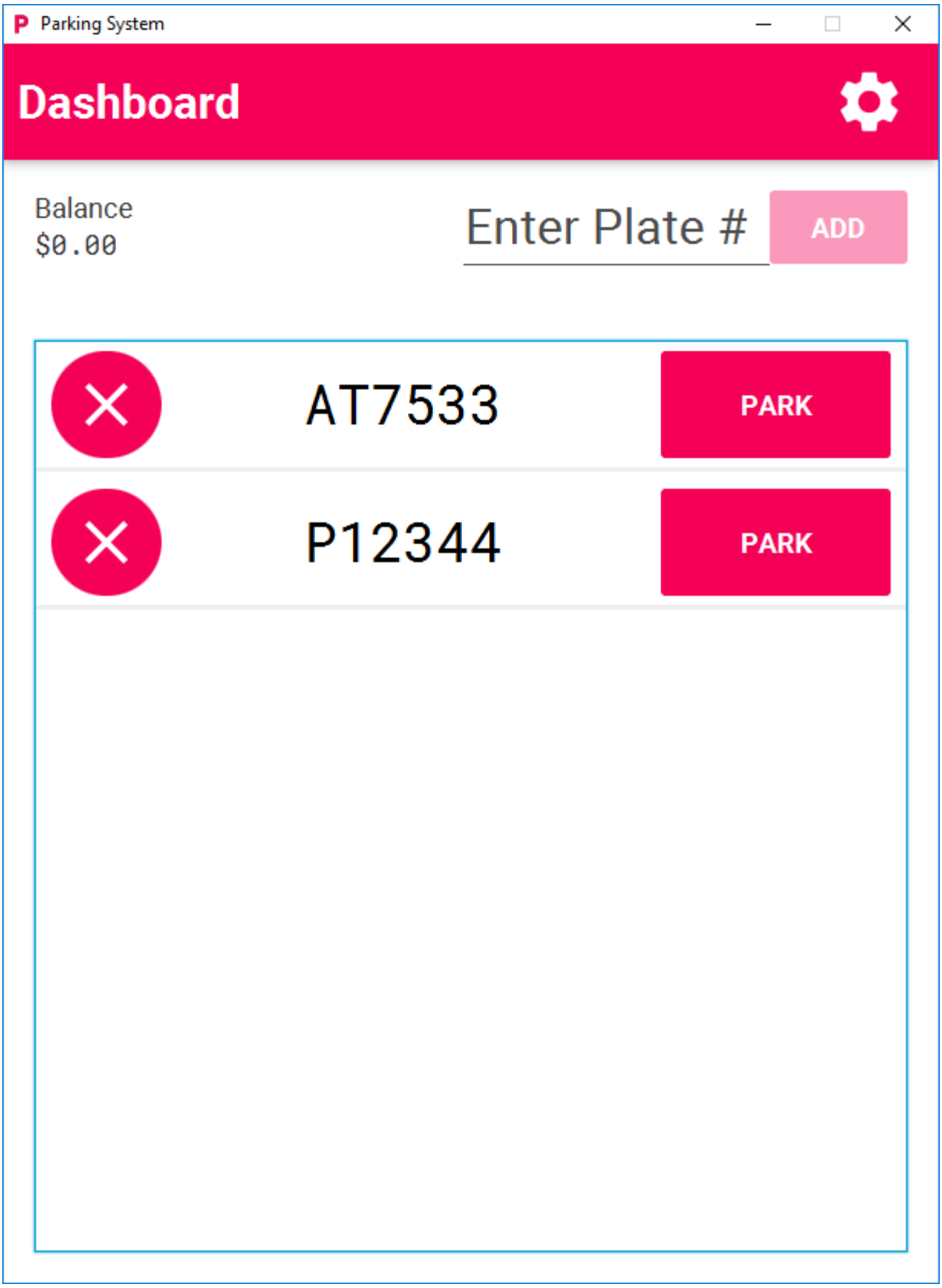
- Add a guest that can park a vehicle without the need to log in.
- Show parking is full and ask if customer wants to get notified when a spot becomes free. Also, allow the customer to reserve the spot some time ahead.
- Customer with lost code pay the day pass rate for VIP.
- Timed session. Members are automatically logged out when idle for a certain amount of time.

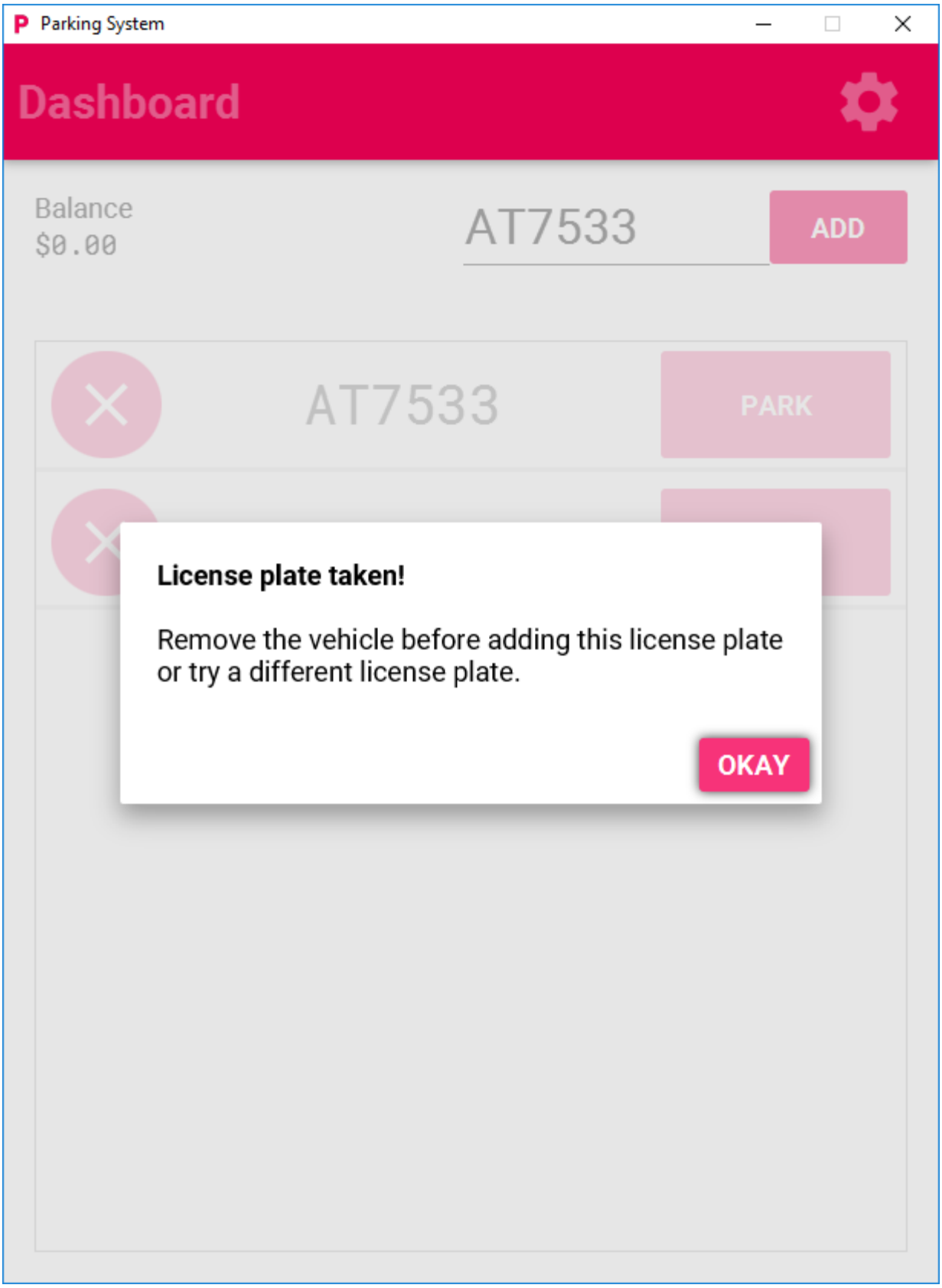

Parking System
Welcome
Member
Register

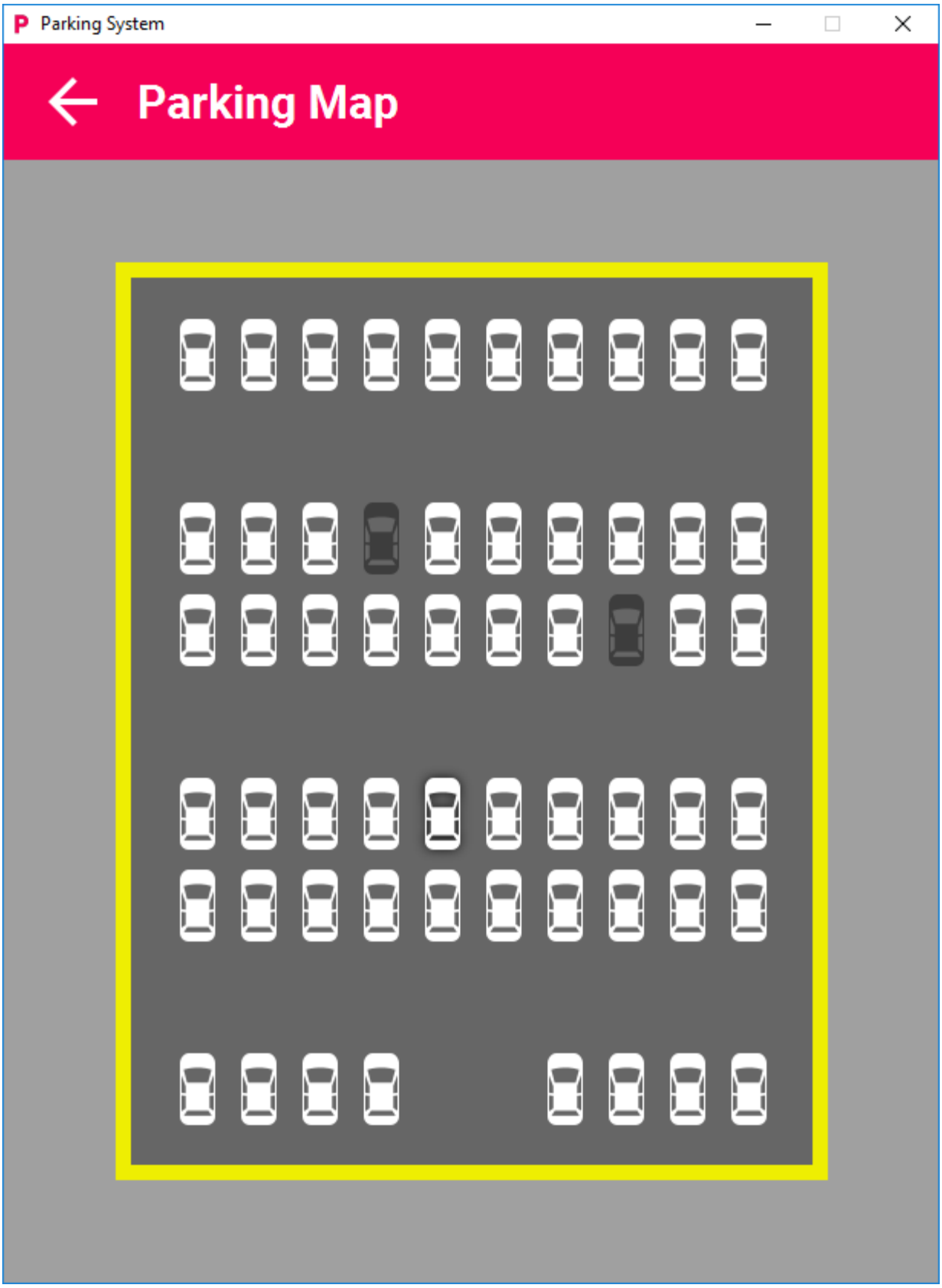

Parking System
Sign up
Username
rami4
Password
Confirm Password
Signup

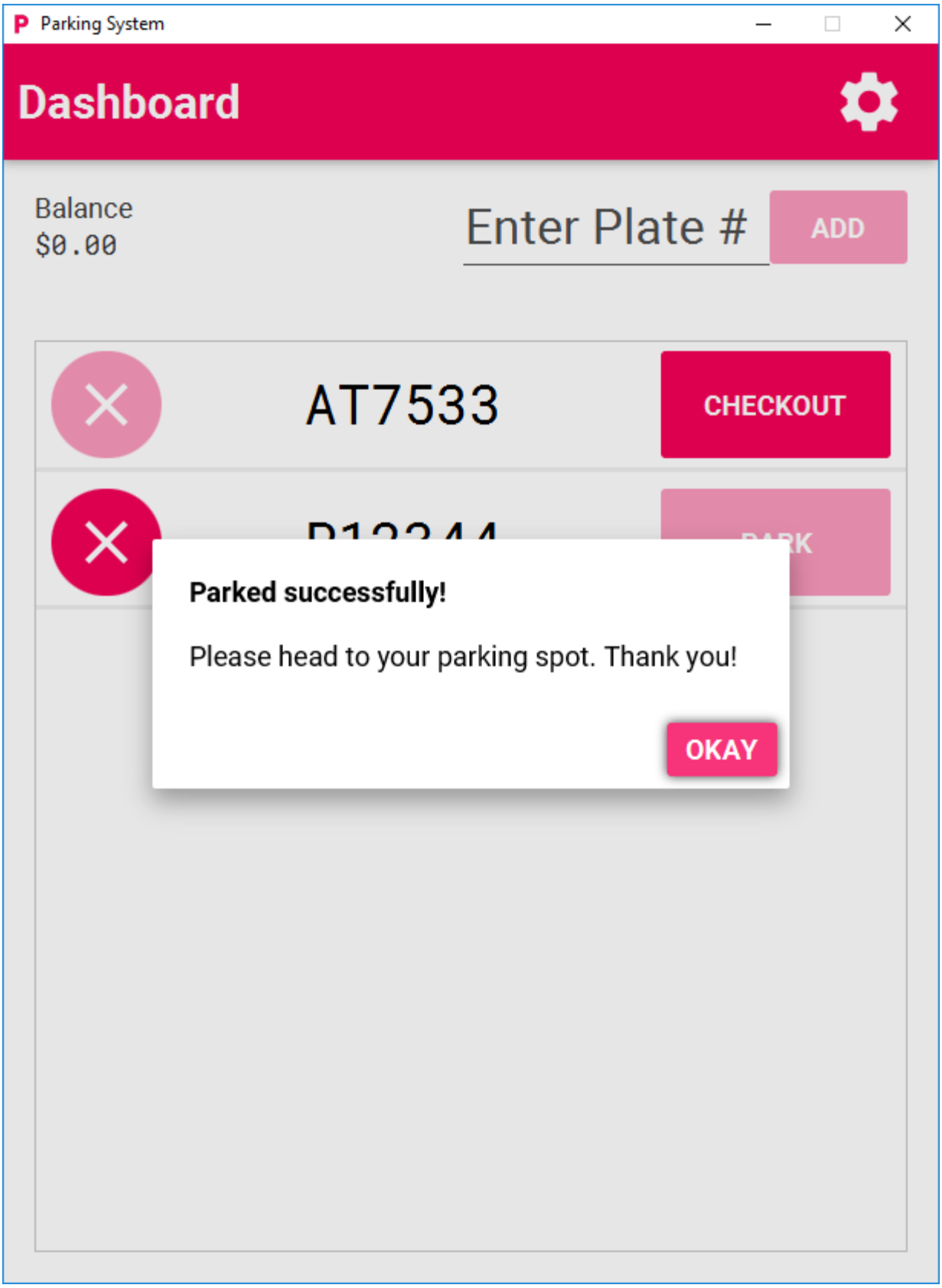

Parking System
Log in
Username
rami4
Password
LOGIN

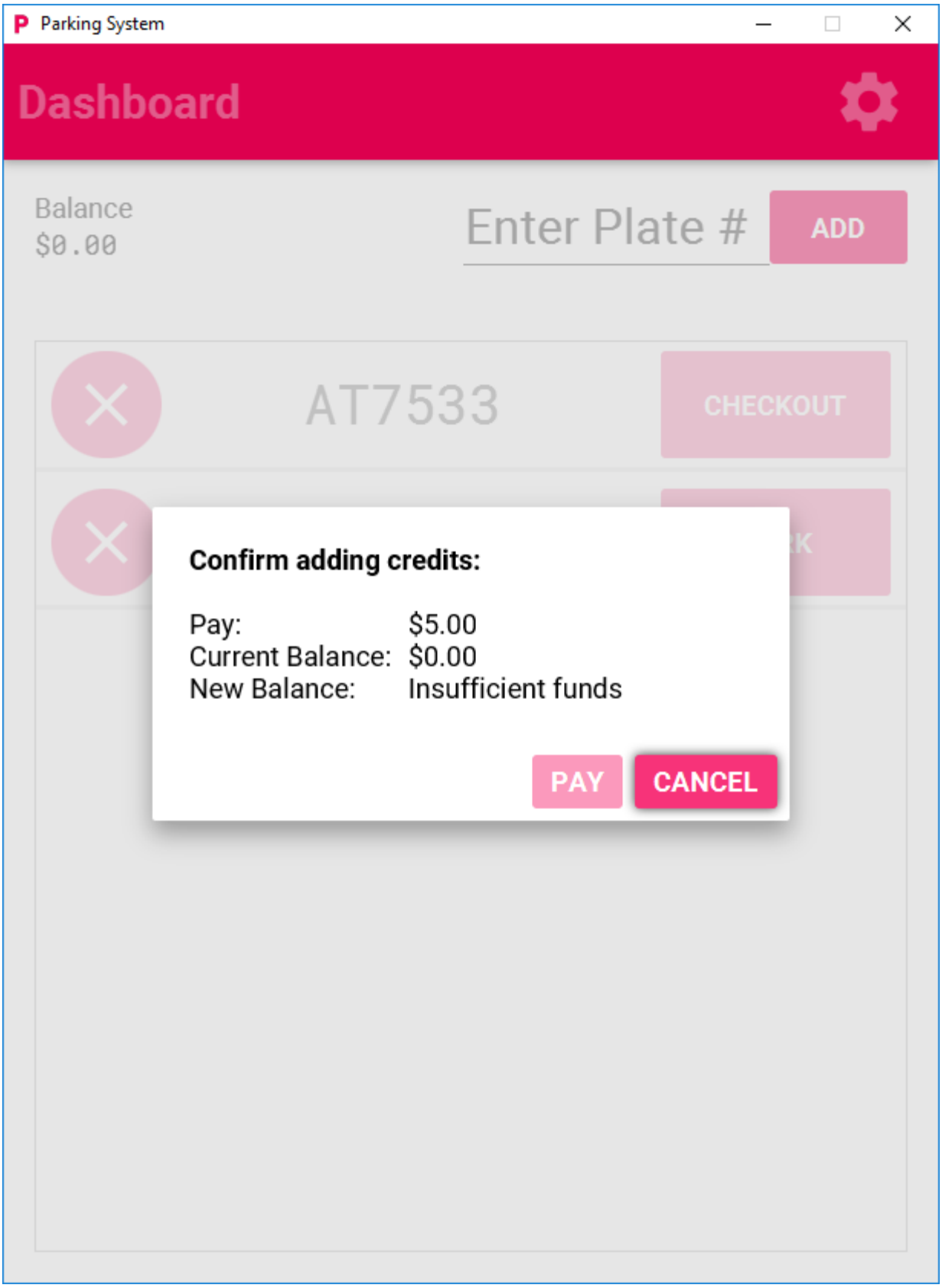

Parking System
Dashboard
Balance
$0.00
AT7533
ADD
Add vehicles to park

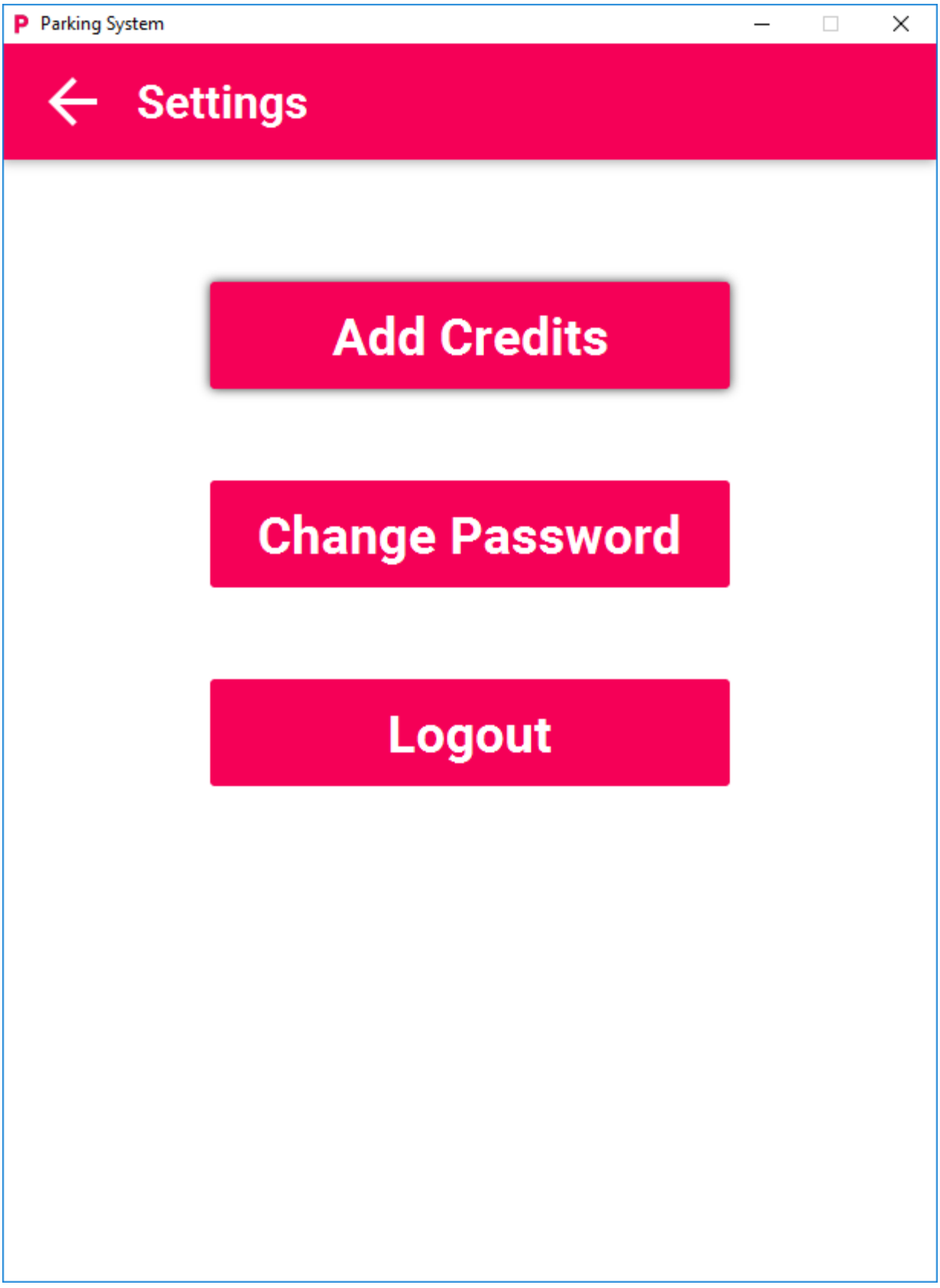

Parking System
Dashboard
Balance
$0.00
Enter Plate #
ADD
AT7533
PARK
P12344
PARK

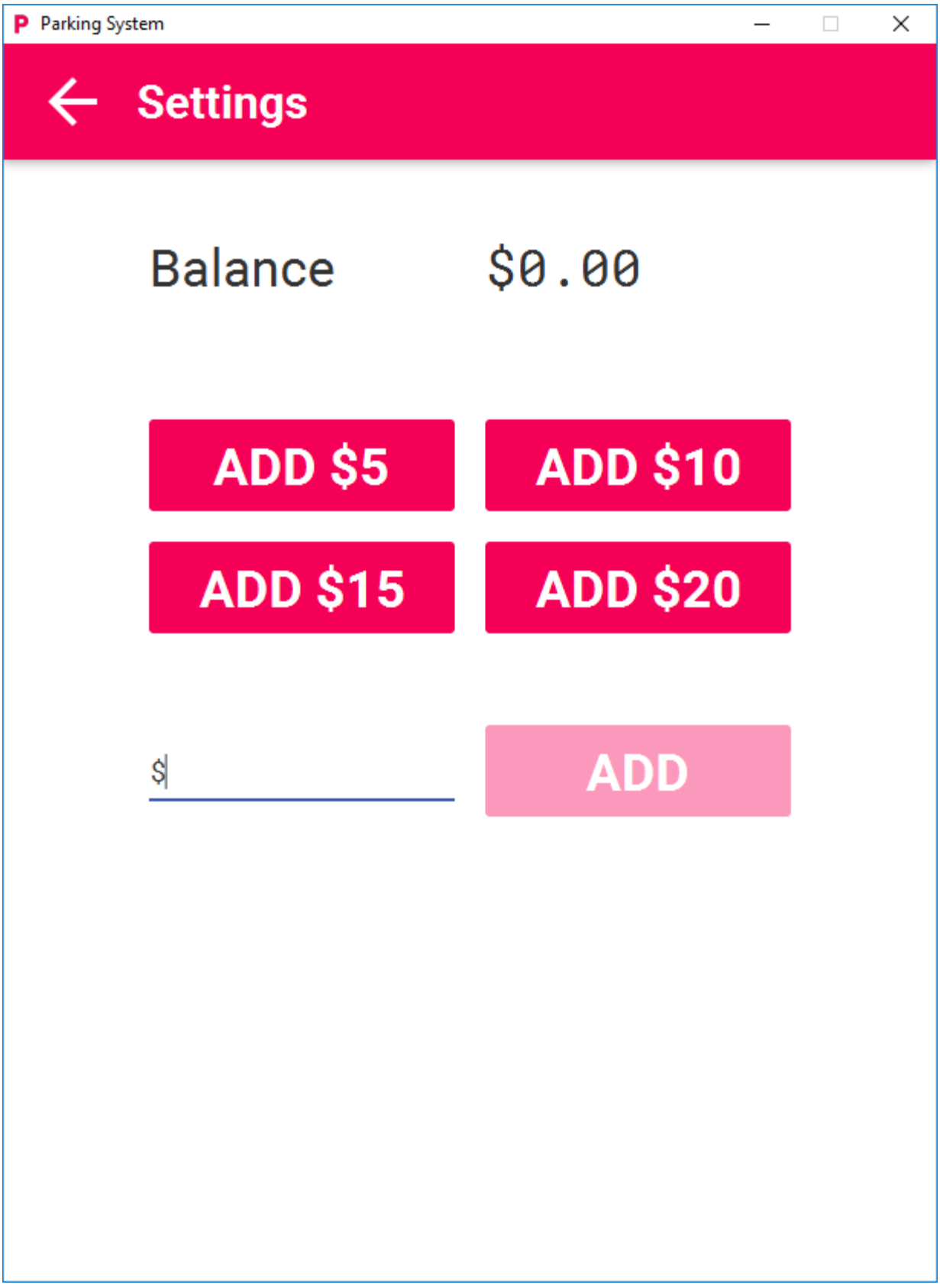

Parking System
Dashboard
Balance
$0.00
AT7533
ADD
AT7533
PARK
License plate taken!
Remove the vehicle before adding this license plate or try a different license plate.
OKAY

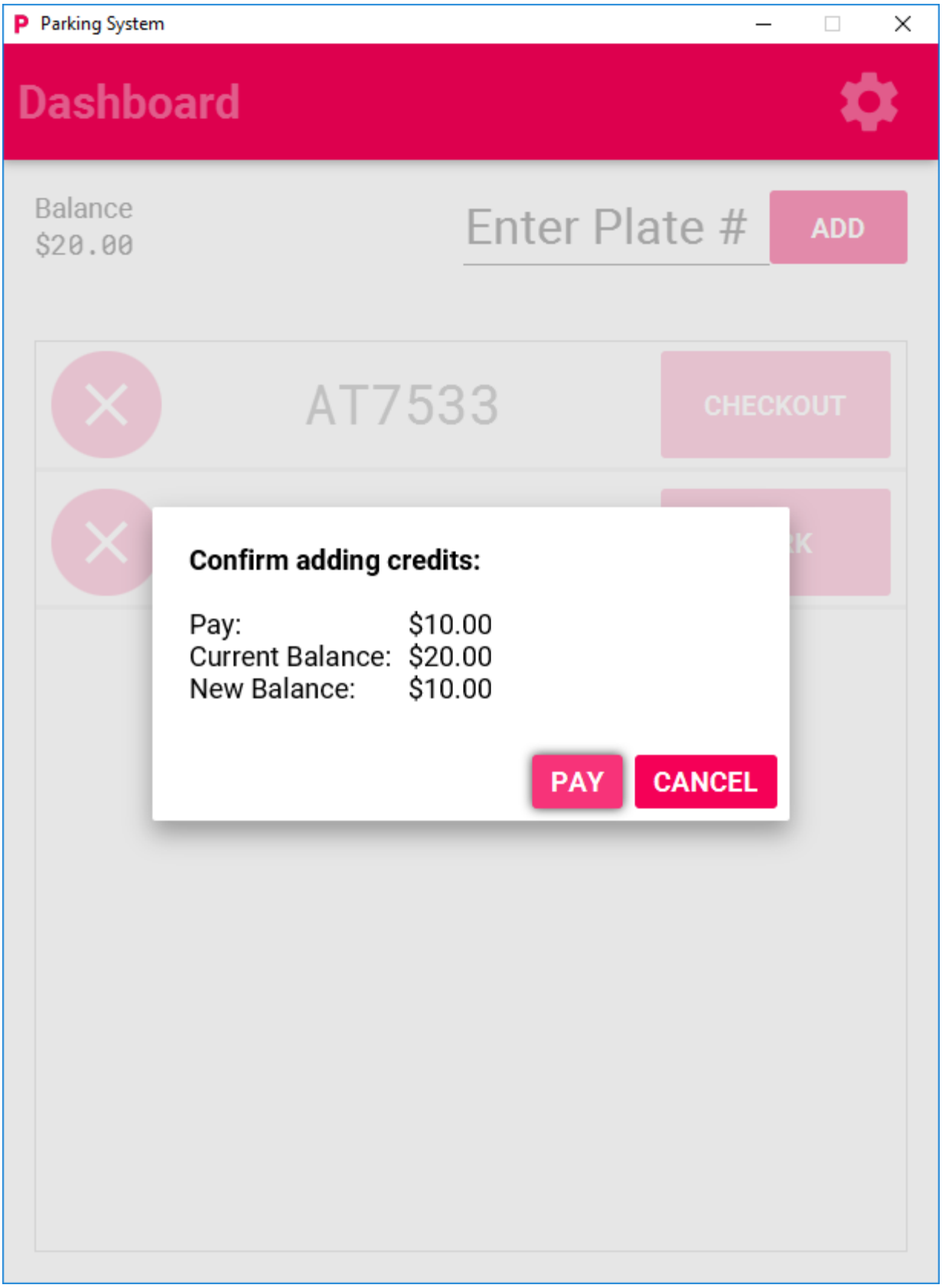

Parking System
Parking Map

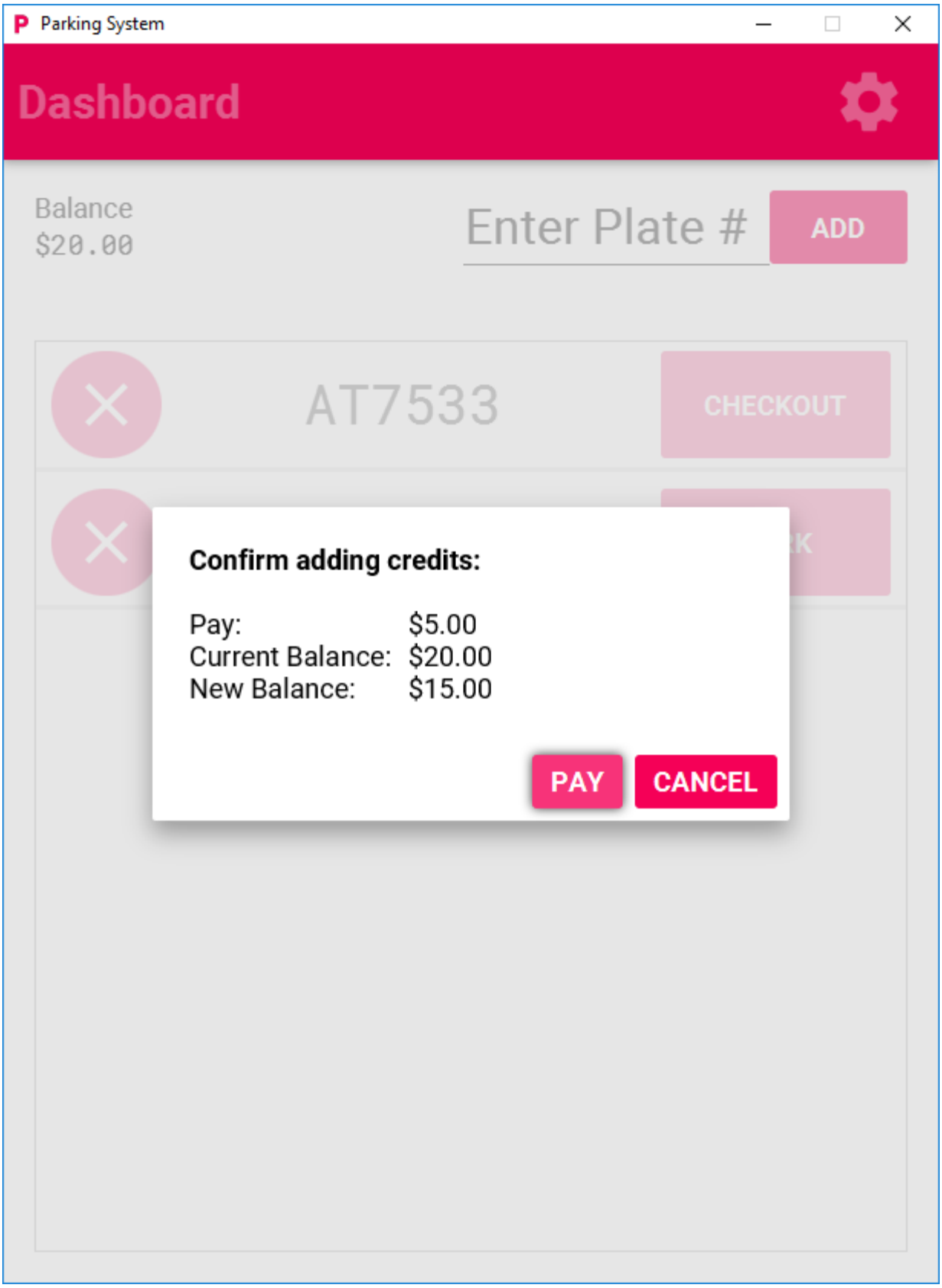

Parking System
Dashboard
Balance
$0.00
Enter Plate #
ADD
AT7533
CHECKOUT
Parked successfully!
Please head to your parking spot. Thank you!
OKAY

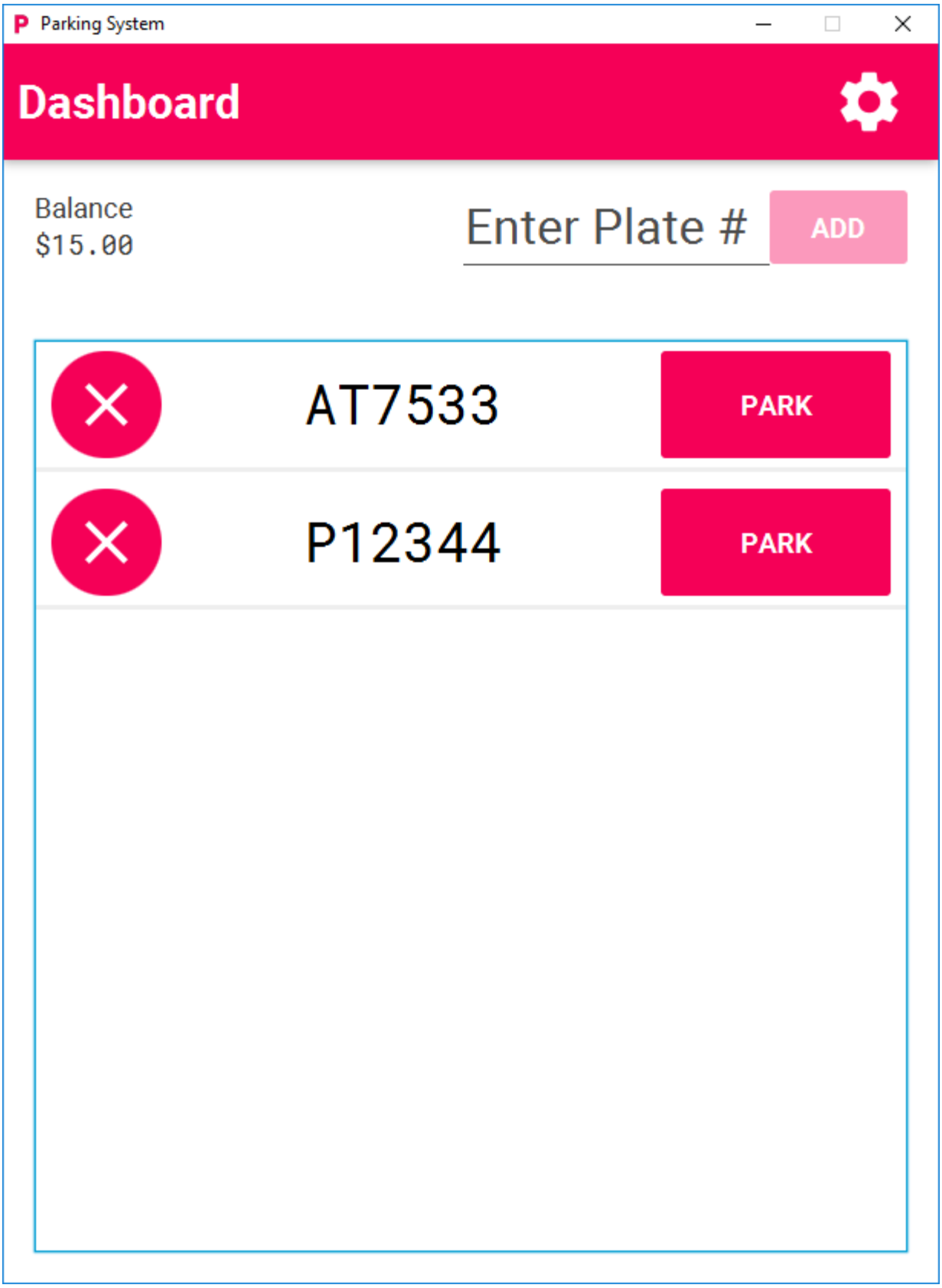

Parking System
Dashboard
Balance
$0.00
Enter Plate #
ADD
AT7533
CHECKOUT
Confirm adding credits:
Pay: $5.00
Current Balance: $0.00
New Balance: Insufficient funds
PAY
CANCEL

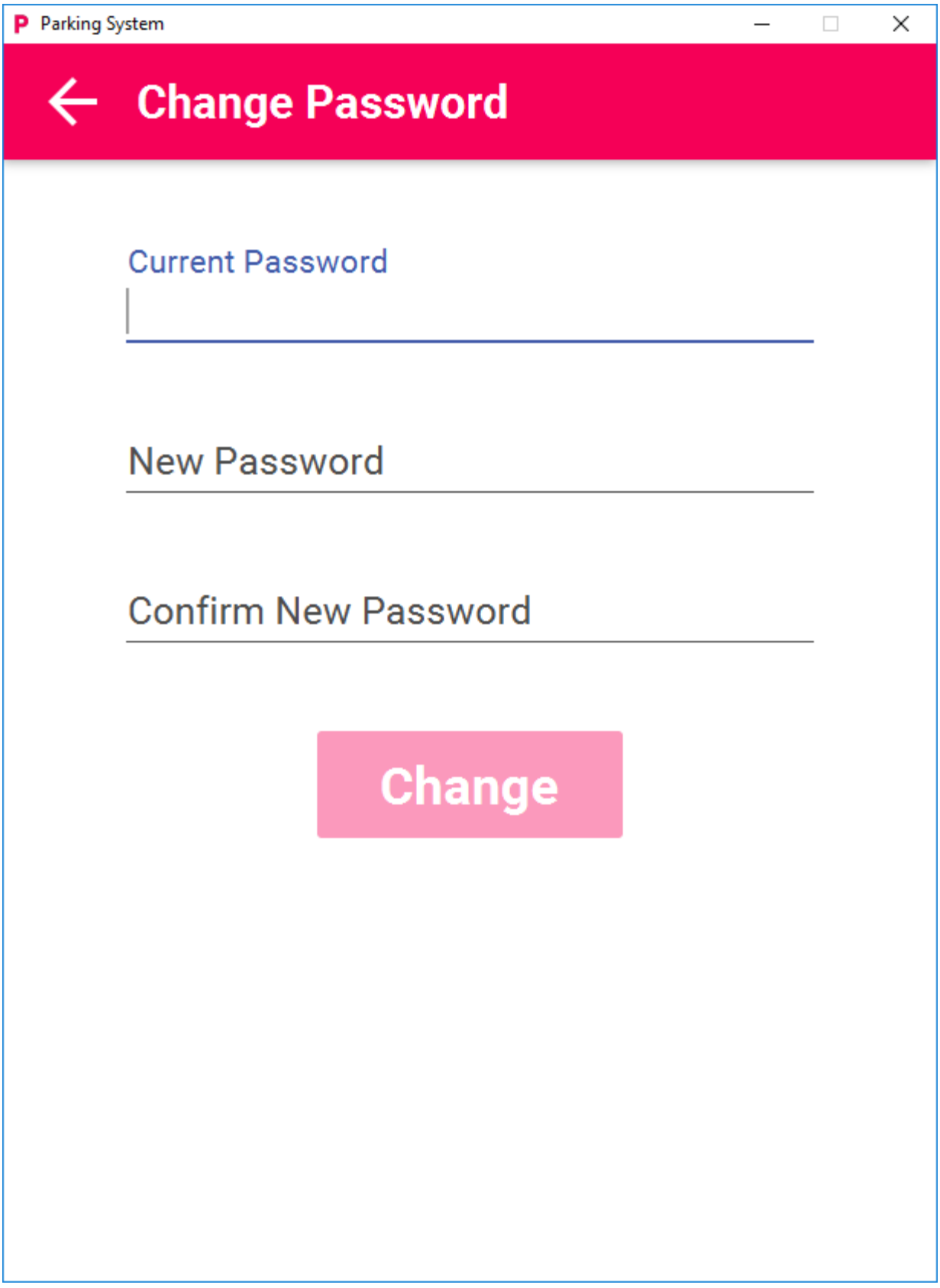

Parking System
Settings
Add Credits
Change Password
Logout


Parking System
Settings
Balance
$0.00
ADD $5
ADD $10
ADD $15
ADD $20
$
ADD

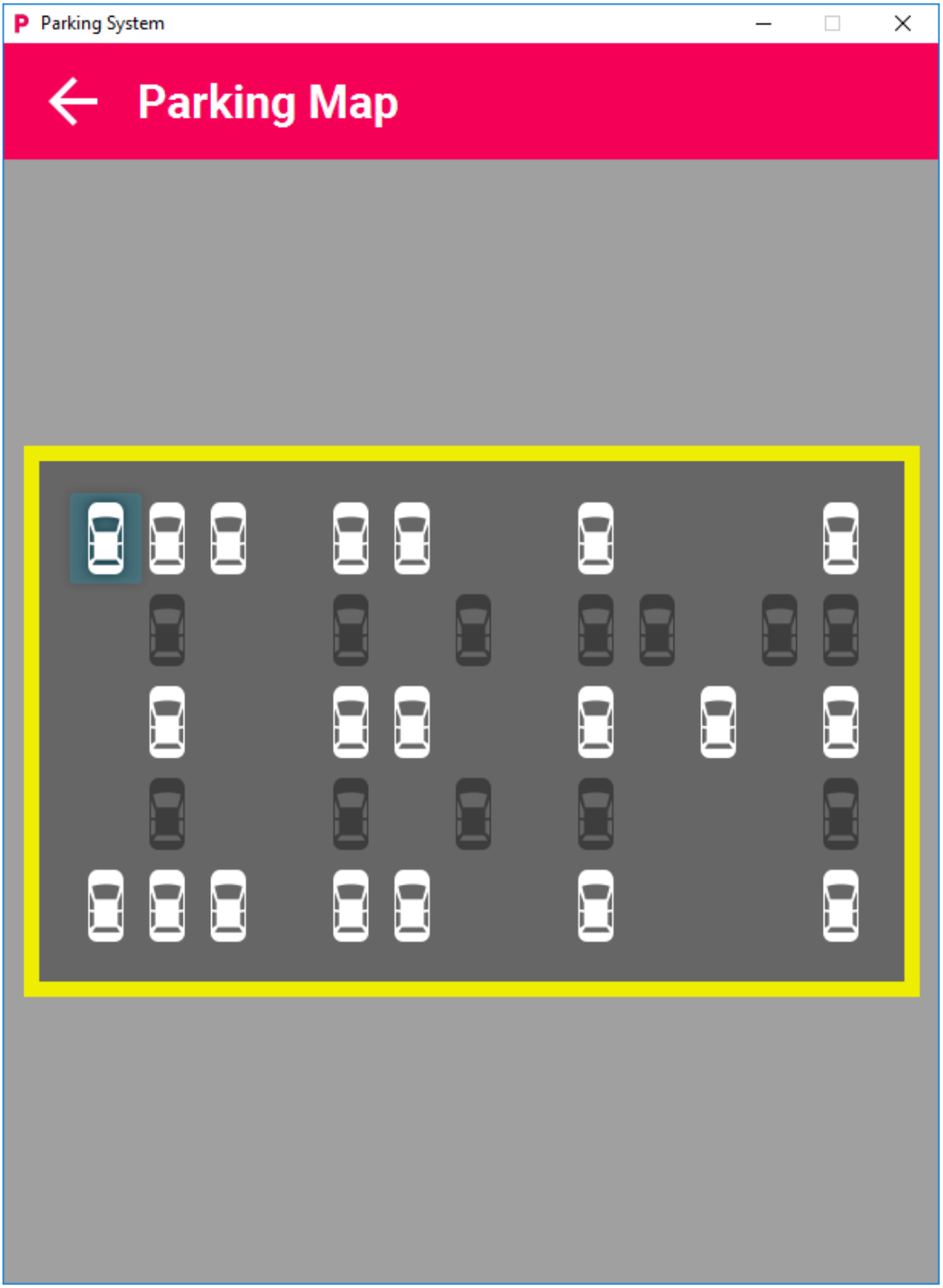

Parking System
Dashboard
Balance
$20.00
Enter Plate #
ADD
AT7533
CHECKOUT
Confirm adding credits:
Pay: $10.00
Current Balance: $20.00
New Balance: $10.00
PAY
CANCEL

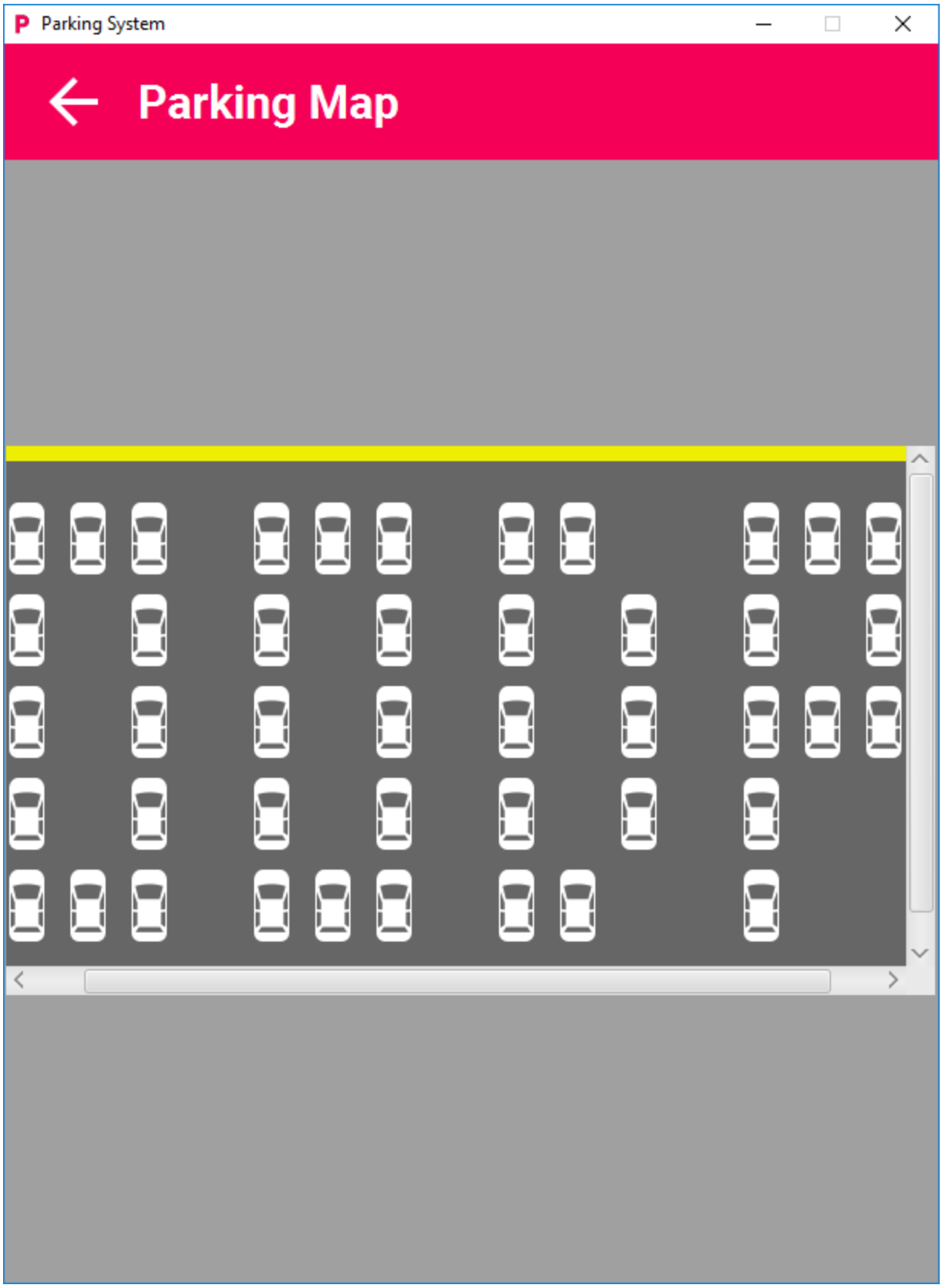

Parking System
Dashboard
Balance
$20.00
Enter Plate #
ADD
AT7533
CHECKOUT
Confirm adding credits:
Pay: $5.00
Current Balance: $20.00
New Balance: $15.00
PAY
CANCEL


Parking System
Dashboard
Balance
$15.00
Enter Plate #
ADD
AT7533
PARK
P12344
PARK


Parking System
Change Password
Current Password
New Password
Confirm New Password
Change

Extra Screenshots




Parking System
Parking Map