**UML Diagrams**

The UML diagram for the Login component. This was made with the singleton pattern. Only one instance can be created.

A screen shot of a login screen

AI-generated content may be incorrect.

The UML diagram for the NotificationList component, which uses the observer pattern.

A diagram of a server

AI-generated content may be incorrect.

The Subject and Observer interfaces have functions that are implemented by the concrete classes. NotificationSubject is a concrete subject that implements Subject. It has the functions for registering and notifying observers. NotificationObserver is a concrete observer that implements Observer and has the notification for updating the interface for the user. When these classes are used in other components, objects of NotificationSubject and NotificationObserver are created, and the NotificationSubject object is observer is set to the NotificationObserver object. When the function that requires a notification is performed, setNotification is used to send a notification to the database.

The UML diagram for UI components, which user the mediator pattern.

A diagram of a construction site

AI-generated content may be incorrect.

The FormDirector interface is the abstract mediator. It has the submit function, which the concrete mediators use. The concrete mediators are all the different app components that involve form submission and need UI components like text entry and buttons. The Widget interface is the abstract colleague that has the basic onClick() and onChange() functions that the concrete colleagues have. The concrete colleagues are TextEntry, NumberEntry, PasswordEntry, Checkbox, Calendar, and Button, which render the various UI components.

The UML diagram for car listing creation, handled by the builder pattern.

A diagram of a car builder

AI-generated content may be incorrect.

Director is the director class. It is instantiated along with a CarBuilder object. AbstractBuilder is the Builder. It is not instantiated or use. The Director object tells the CarBuilder object to build the car list. The CarBuilder object fetches data from the database and creates Car objects with the listed attributes and puts them in the list array, which are arranged and rendered for the user to see. It also puts the cars’ availability information in the dates array, and a copy of this array is used for the corresponding Car’s availability attribute.

The UML diagram for the Verify component, which uses the proxy pattern to allow the user to rent cars.

A diagram of a software system

AI-generated content may be incorrect.

VerificationInterface is the Subject. It is not instantiated. Verification (the real subject) and VerificationProxy (the proxy) inherit from it. The client is the Verify component. In use, an object of VerificationProxy is created in Verify. VerificationProxy has a Verification object in its attributes. When submit is called, it has the verifProxy check the entered password. If it is correct, the proxy calls Verification, whose return allows the user to rent cars.

The UML diagram for the Recovery component, which uses the chain of responsibility pattern to handle password recovery.

A diagram of a bunch of text

AI-generated content may be incorrect.

Handler, the chain of responsibility’s handler, is an abstract class. It has a successor, a function for setting the successor, and an abstract function for handling questions. The four concrete handlers (EmailHandler, Q1Handler, Q2Handler, Q3Handler, and Q4Handler) inherit from Handlers and have their own question handling functions. EmailHandler is the first in the chain. It checks to see if the email address input by the user is in the database. If it isn’t, it returns. Otherwise, it calls its successor, Q1Handler, which checks to see if the user’s answer to the first question is correct, and so on. Recovery is the client. In the submit function, it creates objects of each concrete handlers, sets their successors to set the chain of responsibility, then calls EmailHandler to start the chain. It displays errors depending on what is returned.

**UI Screenshots**

Upon opening the site, users will be taken to the login page.

A screenshot of a computer

AI-generated content may be incorrect.

They must provide a valid email address and password

A screenshot of a login form

AI-generated content may be incorrect.

If they do not have accounts, they can sign up. They cannot use email addresses that are already taken. They will also be required to confirm their passwords. The user must fill out every field.

A screenshot of a computer screen

AI-generated content may be incorrect.

The user can recover their password if they provide the correct answers to the security questions. Because this functionality was implemented using the Chain of Responsibility pattern, the email address and answers are checked one by one. When an incorrect answer is found, only that one will be marked as incorrect even if the next ones are incorrect.

|  |  |
| --- | --- |
| A screenshot of a computer screen  AI-generated content may be incorrect. |  |

When every answer is correct, the user will be shown their password.

A screenshot of a computer screen

AI-generated content may be incorrect.

Upon logging in successfully, the user will be shown a home page. The navigation bar in the top right has links to pages will different functionality.

A screenshot of a computer

AI-generated content may be incorrect.

Users can host cars for rental. They must enter the car’s model, year, mileage, location, price per day, and availability dates. All fields must be filled.

A screenshot of a computer

AI-generated content may be incorrect.

For entering availability, the user can select a date from the calendar. The user can also delete selected dates in case of a mistake.

|  |  |
| --- | --- |
| A screenshot of a calendar  AI-generated content may be incorrect. |  |

The selected dates will be shown in order even if they are chosen out of order.

Once the user has listed a car, they will be shown a notification.



A screenshot of a computer

AI-generated content may be incorrect.

Users will be able to browse cars for renting. A screenshot of a computer

AI-generated content may be incorrect.

Before renting a car, they must enter their passwords to verify themselves.

A white background with black text

AI-generated content may be incorrect.

The user can search for cars by model, location, or availability.

|  |  |
| --- | --- |
| A screenshot of a search box  AI-generated content may be incorrect. | A screenshot of a search box  AI-generated content may be incorrect. |
| A screenshot of a search engine  AI-generated content may be incorrect. | |

The user can click on a car to view its details.

A white rectangular object with black lines

AI-generated content may be incorrect.

The user can choose the dates the want and rent a car. Their account balance will be automatically reduced.

A screenshot of a computer

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

The renter will be notified.

A green and white rectangle

AI-generated content may be incorrect.

The owner of the car will also be notified.



Once a car has been rented, the dates on which it has been rented will not be listed.

A screenshot of a computer screen

AI-generated content may be incorrect.

The user cannot rent their own cars.

A screenshot of a computer screen

AI-generated content may be incorrect.

The user can view the details of cars they have booked or listed for rental.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a car

AI-generated content may be incorrect.

For their own cars, the user can edit the attributes. If the car has not been rented on any date, it can be deleted entirely. The user must enter a value to edit.

A screenshot of a computer

AI-generated content may be incorrect.

Cars and dates cannot be deleted if they are rented.

A screenshot of a computer

AI-generated content may be incorrect.

If the user deletes or edits a car, they will be notified.



On the messaging page, users will be shown their existing conversations (if any) and input fields for starting a new one.

A screenshot of a computer

AI-generated content may be incorrect.

The user must enter both a valid email address other than their own and a message to send a message.

A white rectangular object with gray lines

AI-generated content may be incorrect.

The receiver of the message will be notified.

A white rectangular sign with black text

AI-generated content may be incorrect.  
Users can reply to messages and continue conversations.

A screenshot of a chat

AI-generated content may be incorrect.

Users will be notified whenever they are messaged.

A close-up of a white panel

AI-generated content may be incorrect.

**Database Schema Description**

List of tables and their columns:

* **availability:** contains records of car availability
  + **CID:** the car’s ID (foreign key)
  + **date:** the date on which the car is available
  + There is a record for each availability date
* **cars:** contains records of cars listed for rental
  + **CID:** the car’s ID (primary key)
  + **UID:** the car’s owner’s ID (foreign key)
  + **owner:** the car’s owner’s name
  + **model:** the car's model
  + **year:** the car’s year
  + **mileage:** the car's mileage
  + **location:** the car’s rental location
  + **price:** the price of renting the car for one day
* **messages:** contains messages sent between users
  + **datetime:** the timestamp of the message
  + **senderEmail:** the email address of the sender (foreign key)
  + **receiverEmail:** the email address of the receiver (foreign key)
  + **message:** the contents of the message
* **notifications:** contains notifications sent to users
  + **datetime:** the timestamp of the notification
  + **UID:** the receiver of the notification (foreign key)
  + **notification:** the contents of the notification
* **rents:** records of car rentals
  + **CID:** the rented car’s ID (foreign key)
  + **UID:** the renting user’s ID (foreign key)
  + **date:** the date on which the car is rented
  + Each rented date has its own record
* **users:** contains records of registered users
  + **UID:** the user’s ID (primary key)
  + **name:** the user’s full name
  + **email:** the user's email address
  + **password:** the user's password
  + **secq1:** the user's answer to the first security question
  + **secq2:** the user’s answer to the second security question
  + **secq3:** the user’s answer to the third security question
  + **balance:** the user’s account balance

**References**

* W3Schools
  + <https://www.w3schools.com/html/default.asp>
  + <https://www.w3schools.com/css/default.asp>
  + <https://www.w3schools.com/js/default.asp>
  + <https://www.w3schools.com/php/default.asp>
  + <https://www.w3schools.com/react/default.asp>
* React Responsive Collapsible Section Component (Collapsible)
  + <https://www.npmjs.com/package/react-collapsible>