

In the dashboard (main page), you can see “My policies”. This section displays all policies bought by the logged-in user (so make sure you have stored the logged-in user wallet address in the front-end when the user logs on the system).

Now, you can use the solidity function “viewPolicyByWallet” with the argument wallet address.

|  |
| --- |
| Just modify a little bit from interact.js |

Then an array of policy IDs will be returned. Use the solidity function “viewPolicyByID” to retrieve the policy information iteratively. The following fields are returned, and you only have to display the italic fields.

*policy.policyNumber*,

*policy.departureDate*,

*policy.departureTime*,

*policy.flightNumber,*

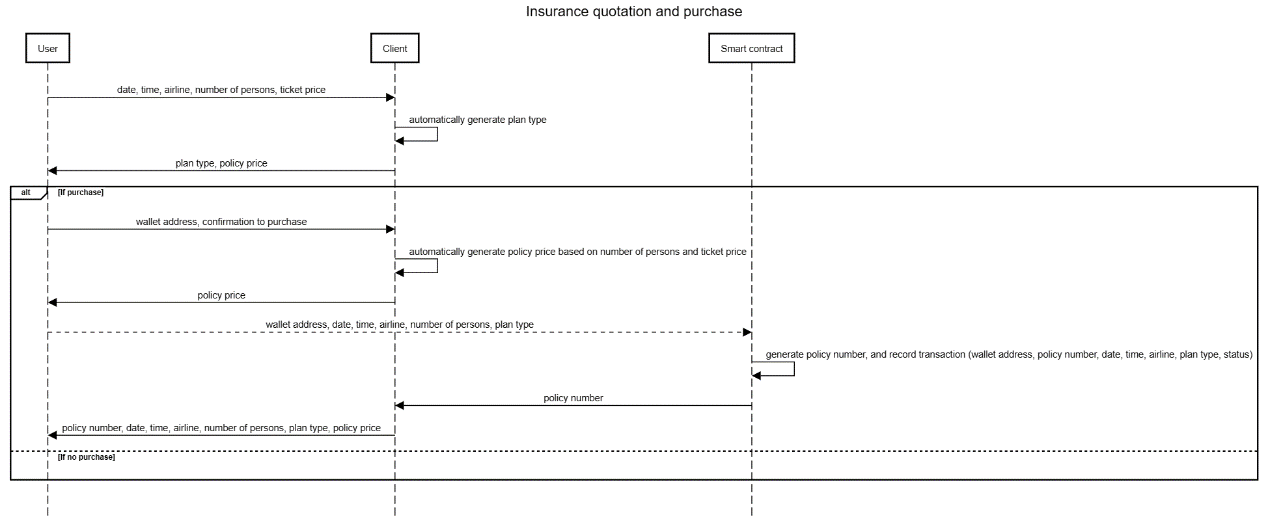
policy.numberOfPersons,

*policy.policyClass*,

*policy.claimStatus* (I forgot to make an amendment. The claimStatus will be either “Pending”, “Claimed” and “Unclaimable”)

Lastly, create a button called “Claim” if the claimStatus is “Pending”.

\* I cannot merge two viewPolicyByxxxxxx functions together because of the limitation of memory allocation in Solidity. So, it may look a little bit clumsy to call two functions. But, it’s not very hard to implement in JavaScript – just take it easy.



In the dashboard (main page), you can see “Secure your next flight”. This section is a form.

There are a couple of fields, each corresponding to a data/time type: date of departure is a data/time. departure time is a time. number of persons is a positive integer. ticket price is a positive floating point number. The plan is auto-generated (Basic if ticket price < 400, otherwise Premium).

When the user clicks “Check out”, show the confirmation page – just show the inputted data. For policy premium, use a simple if-then-else statement to calculate it based on the policyClass.

If the user clicks “Confirm”, you use the function “createPolicy”

uint256 departureDate,

uint256 departureTime,

string memory flightNumber,

uint8 numberOfPersons,

PolicyClass policyClass

Remember to convert the departureDate and time from date to integer format. There are many online resources. As a side note, ticket price is not needed, because we are going to store the policyClass – **0** for Basic and **1** for Premium.

If the function is called successfully, just show the “Your insurance policy application is completed”. It is 90% similar to the “Confirmation” page but the title is different. So, you may consider storing the data in the front-end when the user clicks “Check out” and paste them back here.

Return to the main page if “Return” is clicked.

A diagram with text and arrows

Description automatically generated with medium confidence

In the dashboard (main page), you have created the “Claim” buttons. When the users click it, fetch the delay information from the server. The server should retrieve the record from the database and return it to the server. The server should then return the record to the client.

|  |
| --- |
| * Remember to install MongoDB * Type npm install to install mongoose (a library) * Use the URL like this:   [localhost:8000/flightDelayInfo?departureDate=20220101&departureTime=1200&airline=CX730](http://localhost:8000/flightDelayInfo?departureDate=20220101&departureTime=1200&airline=CX730) |

The client should process the database data a little bit, based on the following rules:

If there is such a record, and the delay time is 3-6 hours, isSmallDelay = True, isLargeDelay is False and isValidClaim are True.

Else if there is such a record, and the delay time is 6 hours, isSmallDelay = False, isLargeDelay = True and isValidClaim is True.

Else

All three fields are false

After that, call the solidity “claim” function with the parameters policyNumber, isSmallDelay, isLargeDelay and isValidClaim. The user will be redirected to the “Your claim application is completed” page, with the details. You can retrieve departureDate, departureTime, airline, numberOfPersons from solidity function “viewPolicyByID” (go back to the second page for details), while delay time and claim amount are from the database (you can calculate the claim amount from delay time).

If the “claim” function fails, show the message “Your claim application is declined” with the same data except Delay time and Claim amount.

Return to the main page if “Return” is clicked.