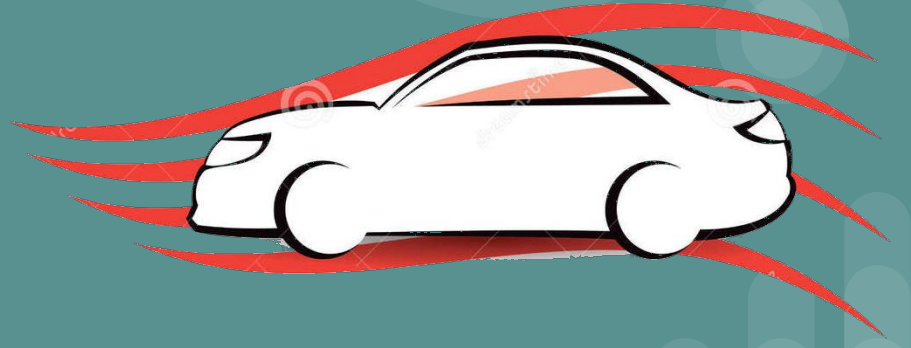


Drivercity

Group 7

Mahima Shrestha
Khatina Haidari
Ahnaf Shahriar Abir



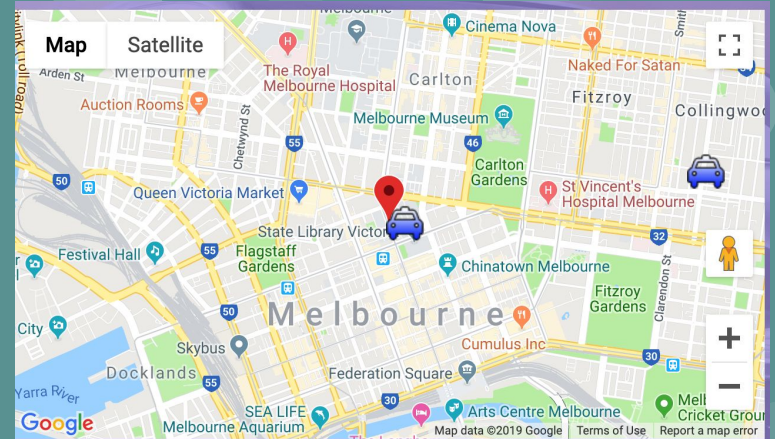
Contents

1. Requirements
2. Meetings
3. Sprint 0
4. Sprint 1
5. Sprint 2
6. Sprint 3
7. Product



Requirements

- Show Current location of the user
- Show available cars nearby the users.
- Get the direction from the user to the car
- Book and Return car
- Payment



Meetings and Challenges

- The meeting were held frequently every 2 days a week.
- Use of Messenger for online communication
- Use of trello for assigning and google drive for sharing tasks
- Challenges:
 - implementing maps page and fetching of data from database
 - Installation of Laravel and Google Cloud
- Used version control (GitHub)



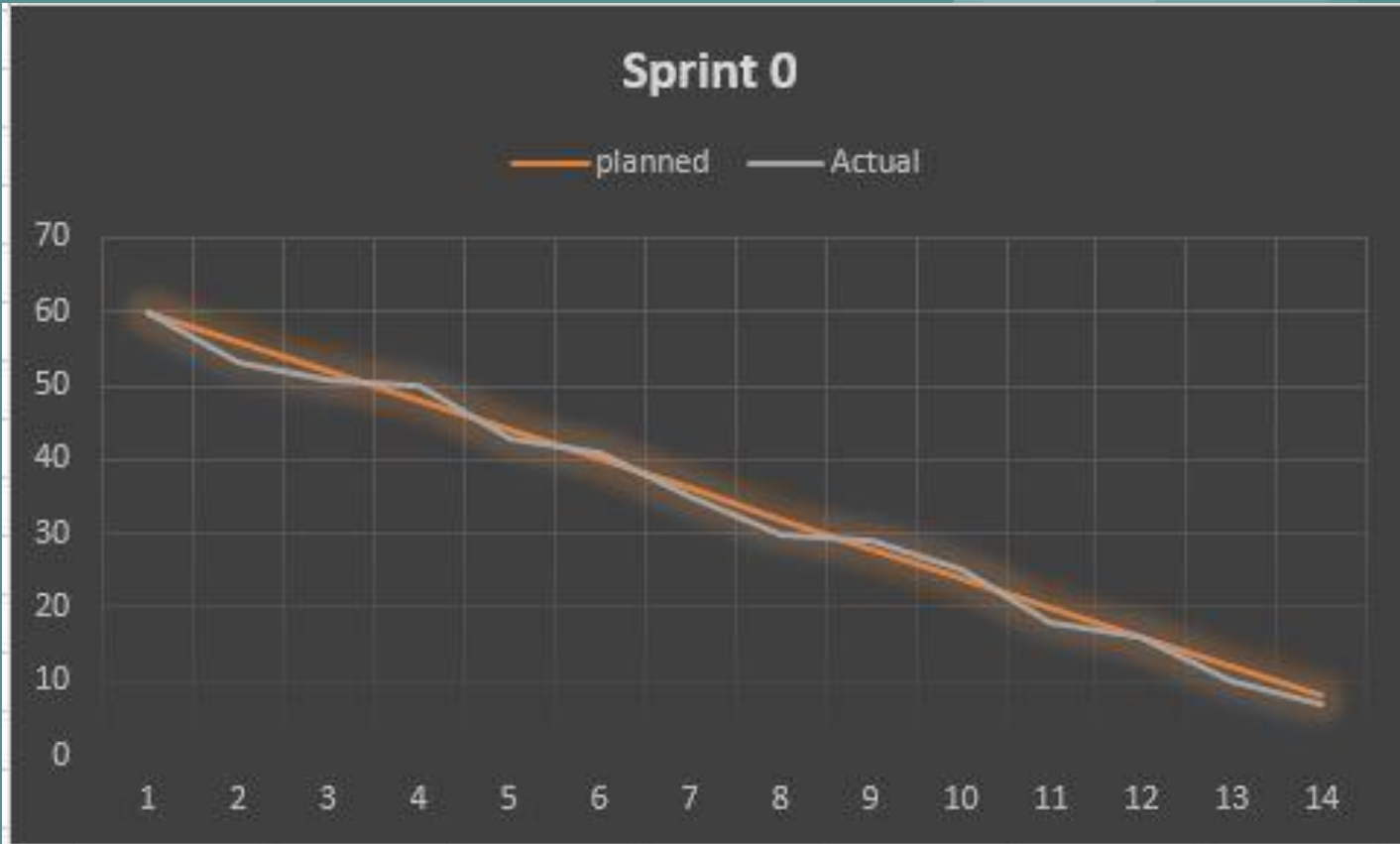
Sprints 0

Goals

- Researched about laravel framework
- Connecting to GitHub
- Draft of project charter.

Story Point: 61 points

Burndown chart



Sprint 0 retro

What went well

Documentation

What went wrong

We underestimated the scope
We didn't break down the tasks

What could be improvised

Understanding the requirements

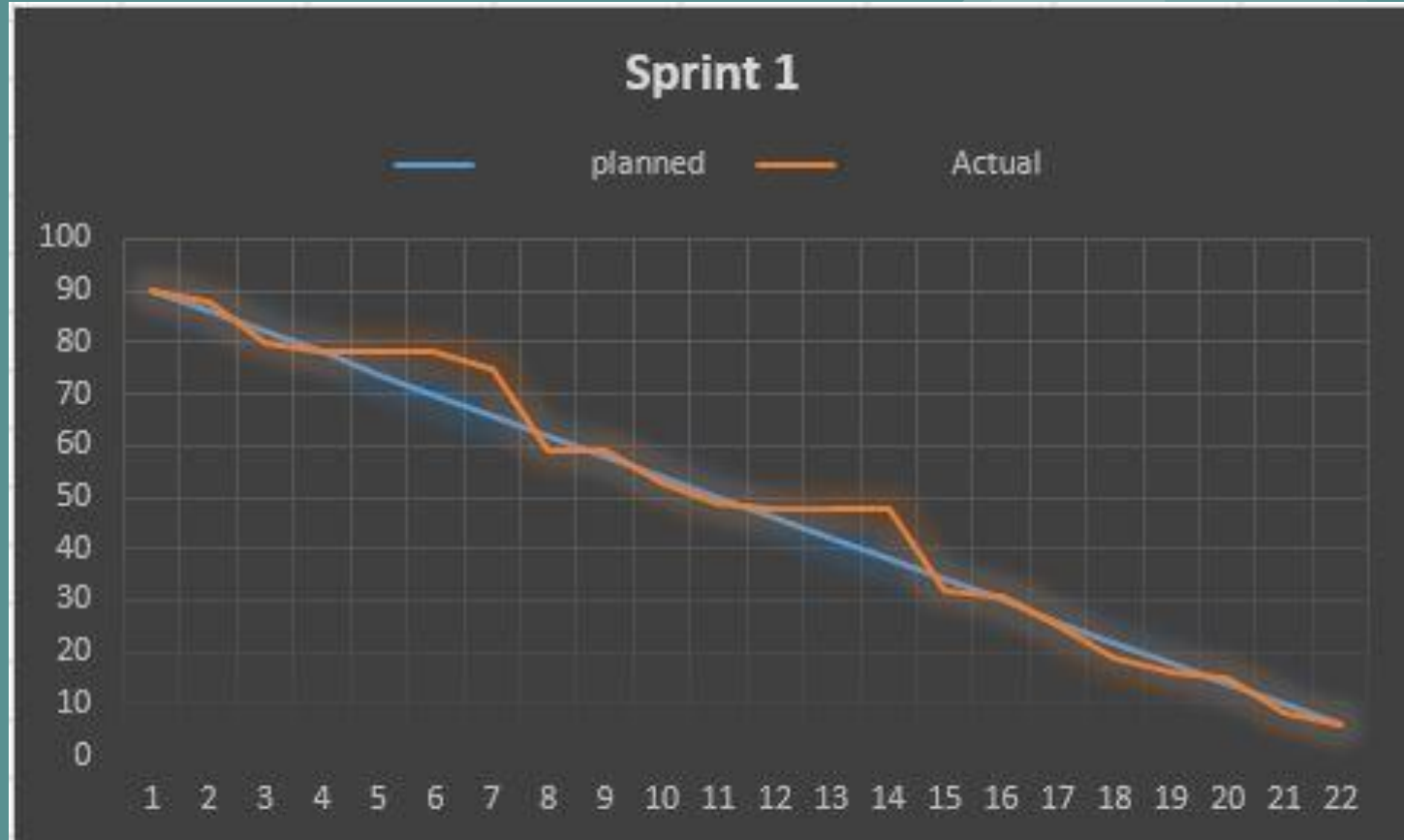
Sprint 1

Goals

- Basic design of the website with home, map, about page
- Making a responsive website.
- Implementing the website with maps and creating a database with tables for users and cars.
- Implementing data in the cars table

Story Point: 98 points

Burndown Chart



Sprint 1 retro

What went well

Responsive website
Basic layout of the website

What went wrong

Installation of Laravel
Setting up GitHub

What could be improvised

Setup an AWS environment
for everyone to work on

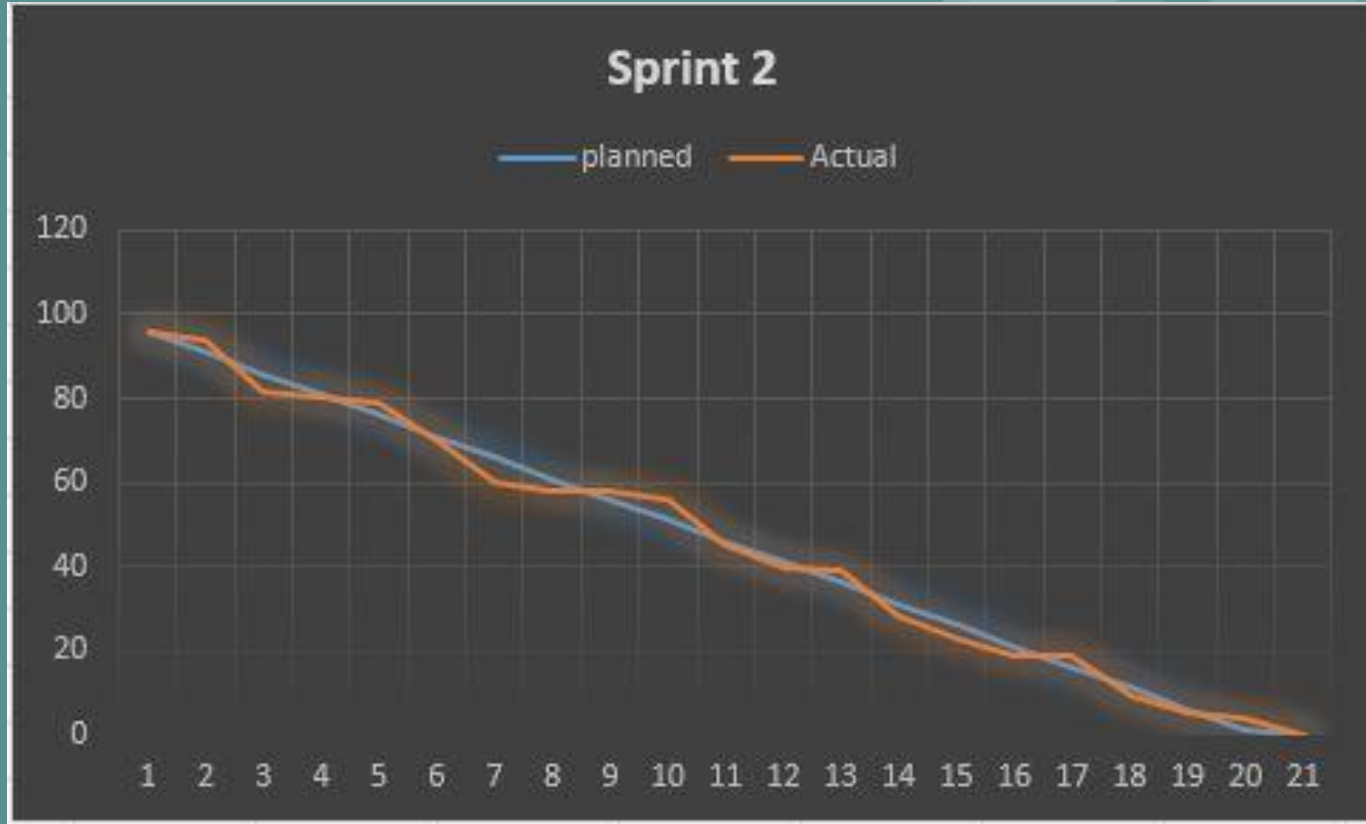
Sprints 2

Goals

- Implementing the maps and cars page
- Adding images in website
- Creating individual car page which fetches data for each car from the database
- Implementing login and registration page
- Documentation

Story Point: 96 points

Burndown Chart



Sprint 2 retro

What went well

Fetching car details from mysql client

What went wrong

Login and registration
Google maps api

What could be improvised

Understand the basics of
Laravel

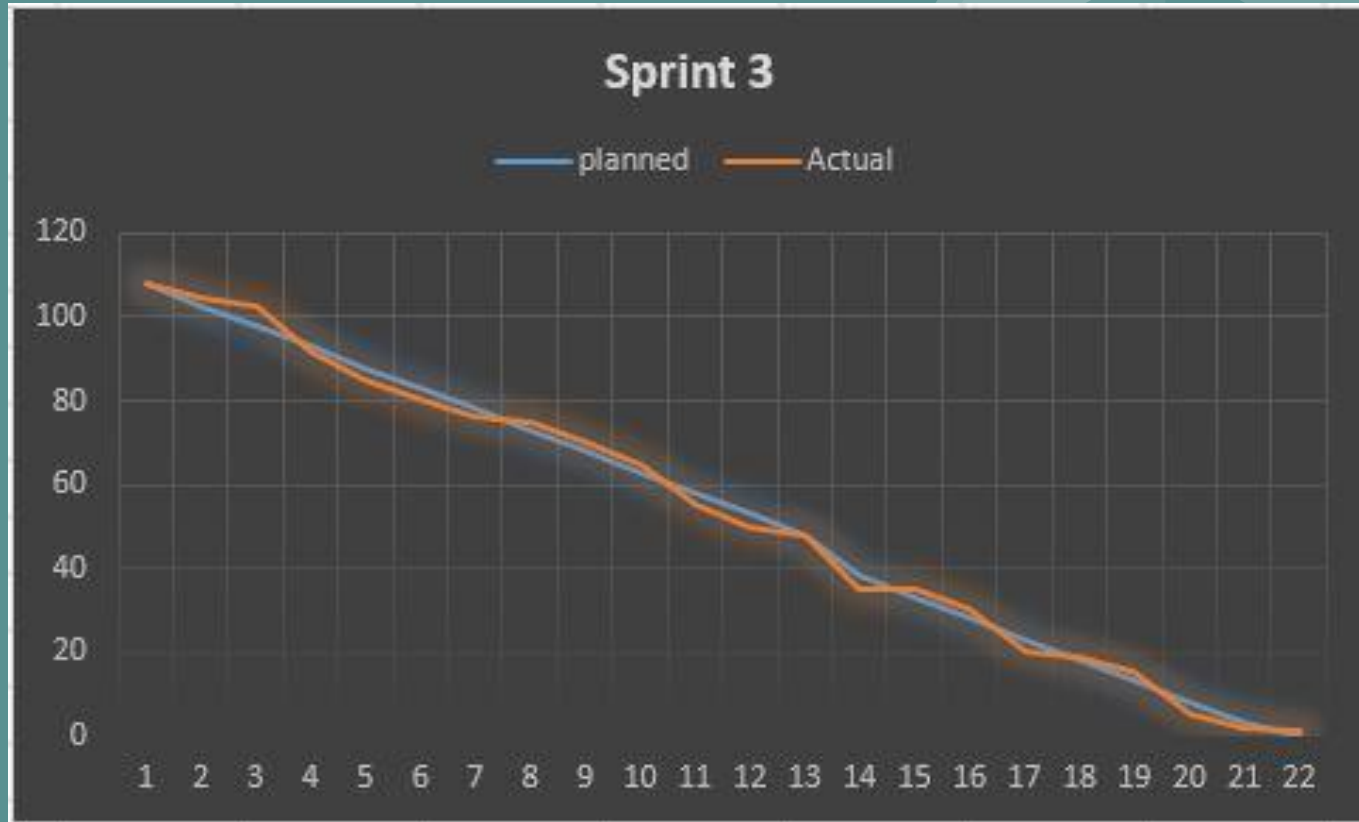
Sprint 3

Goals

- Linking book car page with JS functionality
- Implementing payment option with Stripe and paypal
- Implementing Bootstrap functionality
- Listing of the cars beside the map

Story Point: 108 points

Burndown Chart



Sprint 3 retro

What went well

Bootstrap

What went wrong

Linking javascript and laravel together

Implementation of Stripe

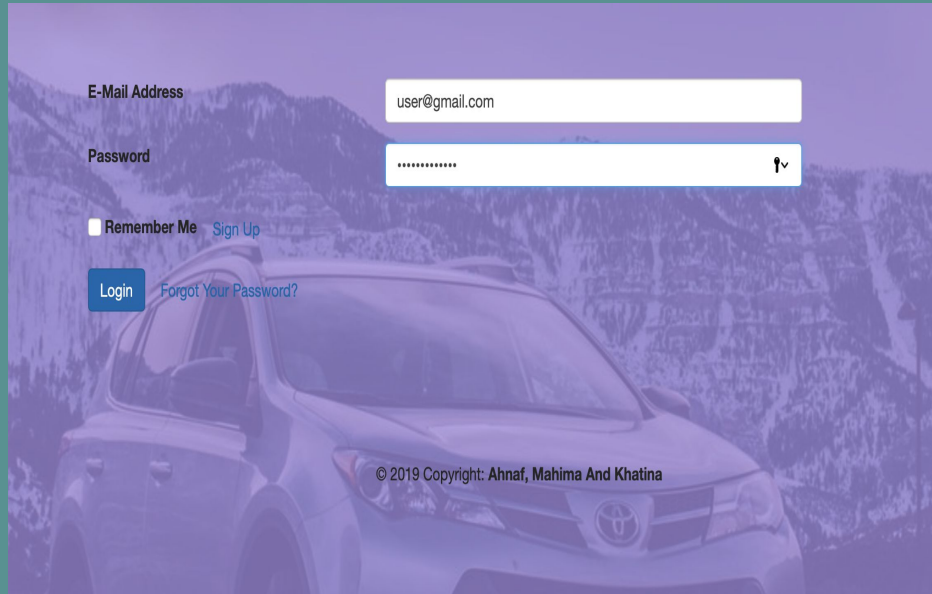
What could be improvised

Research more

Product



Homepage



The login form is overlaid on a purple-tinted image of a Toyota SUV parked in front of a mountain range. The form includes fields for E-Mail Address and Password, a Remember Me checkbox, a Sign Up link, a Login button, and a Forgot Your Password? link. A copyright notice is at the bottom.

E-Mail Address


Password

☐ Remember Me [Sign Up](#)

[Login](#) [Forgot Your Password?](#)

© 2019 Copyright: Ahnaf, Mahima And Khatina

Register



The register form is overlaid on a purple-tinted image of a car. It includes fields for Name, E-Mail Address, Password, and Confirm Password, a Register button, and a copyright notice at the bottom.

Register
Name

E-Mail Address

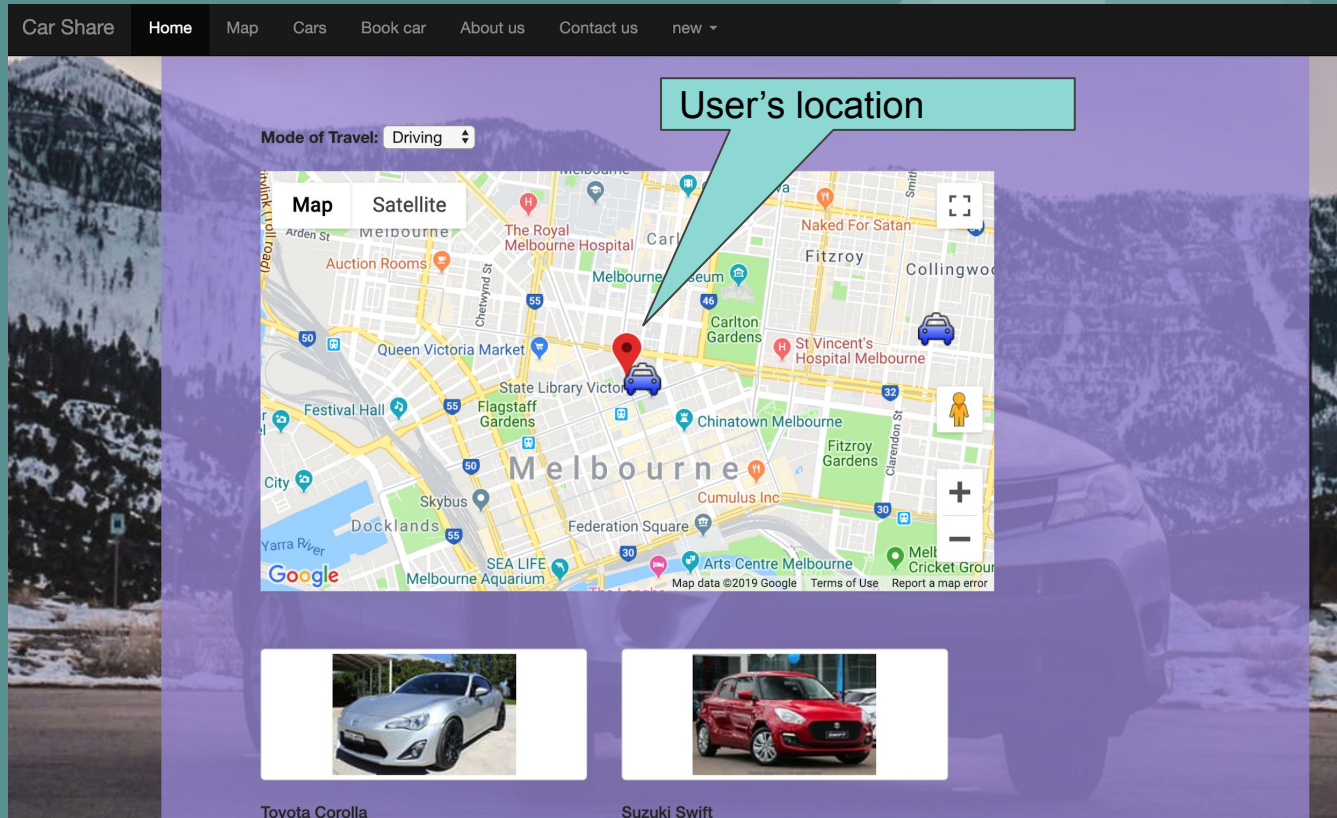
Password

Confirm Password

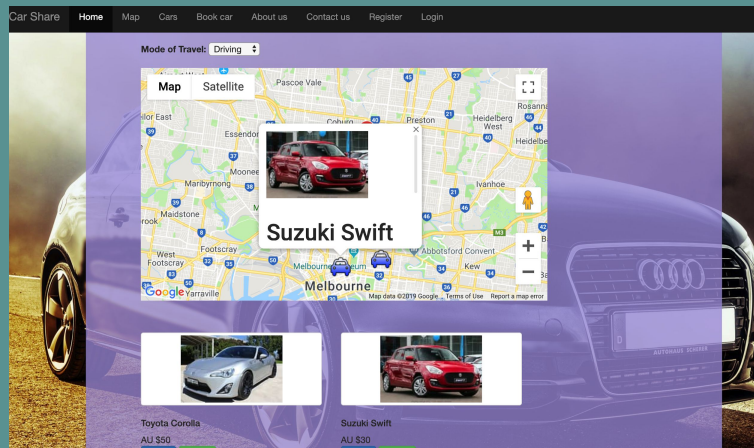
[Register](#)

© 2019 Copyright: Ahnaf, Mahima And Khatina

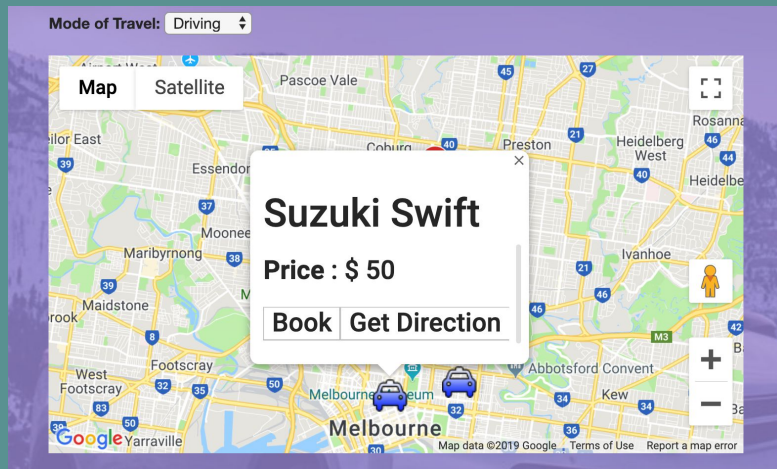
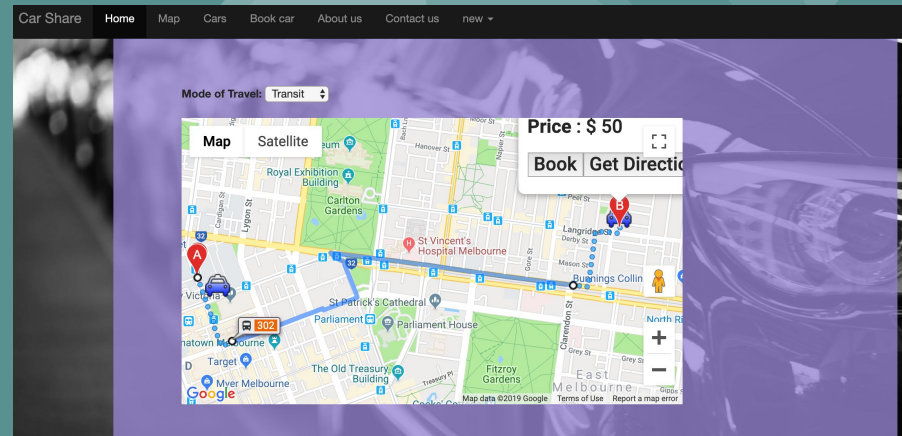
Show User's Current location and list cars nearby



Car details



Direction to car



Book car

Timer starts

Fill up the form within:

9m 49s



Car Details:

Suzuki Swift 2017
Price(per hour): \$ 30

User Details:

User Name

User Name

User Email

user@gmail.com

Phone

Phone

Book

Inserts current
user's details

Booking History

Booked cars History

Booked car: Suzuki Swift 2017

Start Time was: ["2019-05-25 09:52:54"]

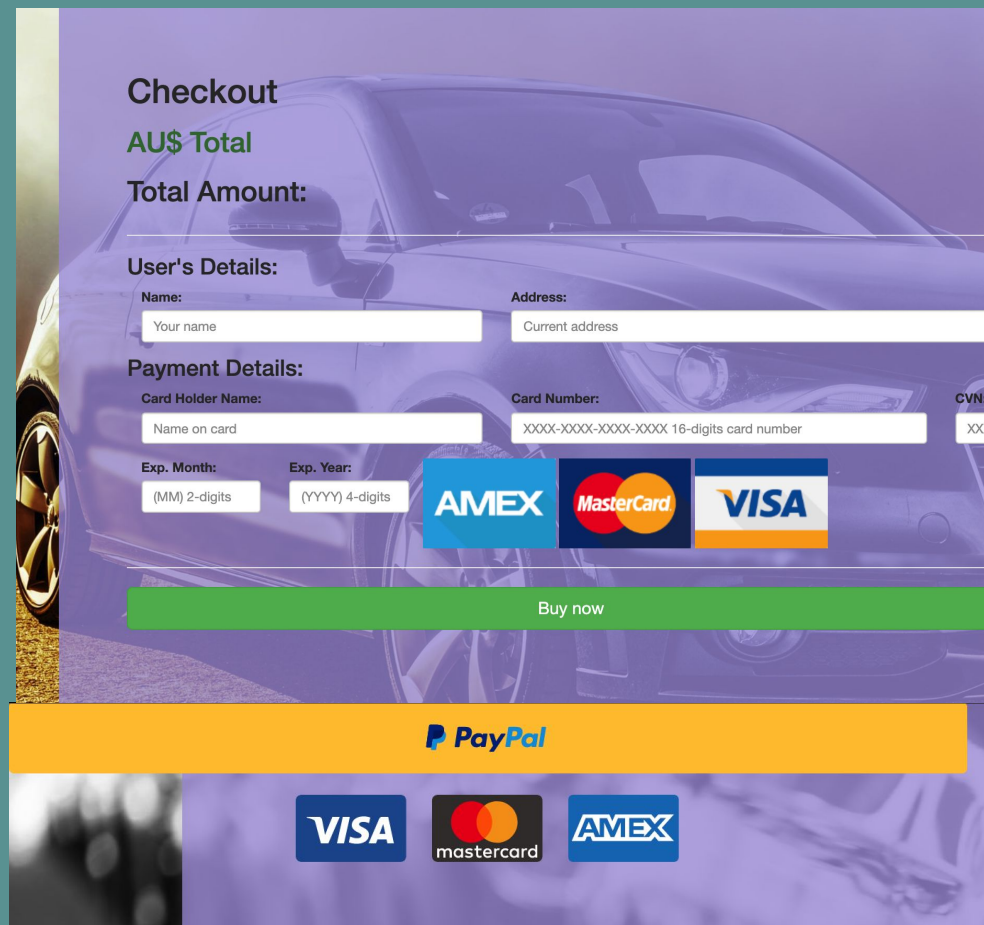
Price: \$30

Calculate total price

Total Price:

Return car

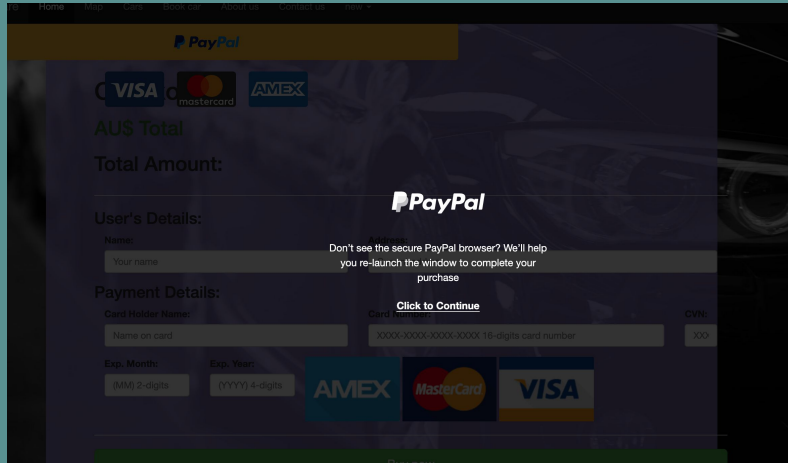
Payment page



The image shows a checkout page for a car purchase. The background is a purple-tinted image of a silver sports car. The form is white and contains the following sections:

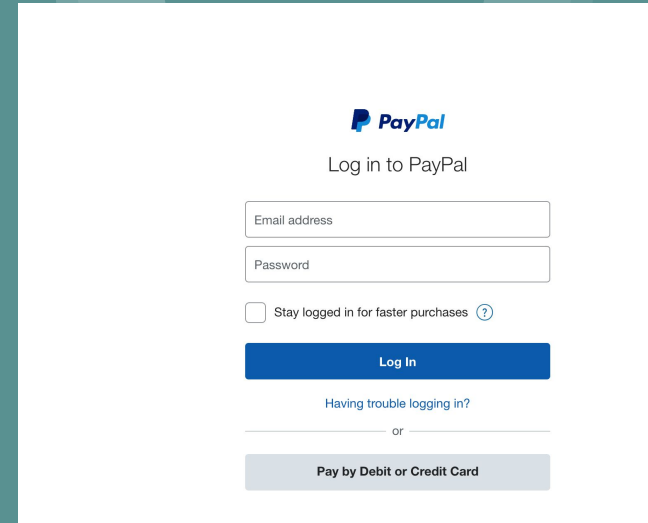
- Checkout**
- AU\$ Total**
- Total Amount:**
- User's Details:**
 - Name:** Input field with placeholder "Your name"
 - Address:** Input field with placeholder "Current address"
- Payment Details:**
 - Card Holder Name:** Input field with placeholder "Name on card"
 - Card Number:** Input field with placeholder "XXXX-XXXX-XXXX-XXXX 16-digits card number"
 - CVN:** Input field with placeholder "XXX"
 - Exp. Month:** Input field with placeholder "(MM) 2-digits"
 - Exp. Year:** Input field with placeholder "(YYYY) 4-digits"
- Payment Options:** Three logos: AMEX, MasterCard, and VISA.
- Buy now** button
- PayPal** logo
- Payment Logos:** VISA, mastercard, and AMEX logos at the bottom.

Paypal payment



A screenshot of a website's checkout page with a PayPal overlay. The website header includes links: Home, Shop, Cart, About Us, Contact Us, and Help. The PayPal overlay is semi-transparent and contains the following elements:

- Logos for VISA, Mastercard, and AMEX.
- AUS Total**
- Total Amount:**
- User's Details:**
 - Name:
 - Address:
- Payment Details:**
 - Card Holder Name:
 - Card:
 - CVV:
 - Exp. Month:
 - Exp. Year:
- Logos for AMEX, MasterCard, and VISA.
- A "Click to Continue" button.
- A message: "Don't see the secure PayPal browser? We'll help you re-launch the window to complete your purchase"



A screenshot of the PayPal login page. It features the PayPal logo at the top, followed by the text "Log in to PayPal". Below this are two input fields: "Email address" and "Password". A checkbox labeled "Stay logged in for faster purchases" with a help icon is positioned below the password field. A blue "Log In" button is centered below the checkbox. Below the button, there is a link "Having trouble logging in?" followed by a horizontal line with "or" in the center. At the bottom, there is a light gray button labeled "Pay by Debit or Credit Card".

Thank you

