UBER API Documentation
Group 4

Rides:

Use Case	Function	Description	Inputs/Outputs	Algorithm
1	**API : requestRide** **Type : POST**	Lets the customer request a ride	**Inputs:** Current location **Output:** Route	**1.** User logs in **2.** Receives geo-location of the user **3.** Once route is set, Driver ID and customer ID generated
2	**API : cancelRide** **Type : GET**	Lets the user cancel the requested ride	**Inputs:** Requested ride info **Output:** Confirmation Status	**1.** User selects the requested ride. **2.** Enter Username and Password for user authentication. **3.** If returns true, cancels the ride and if false aborts
3	**API : fareEstimate** **Type : POST**	Gives the estimate of ride from source and destination	**Input:** Source and destination **Output:** Estimate of ride	1. User provides source and destination 2. User selects route preference i.e. Shortest or Economical 3. Outputs the fare estimate
4	**API: contactDriver** **TYPE : POST**	Gives the user Driver information	**Input:** Ride ID **Output:** Driver Info	1. User selects the ride from list of available requested ride or enter Ride ID 2. Returns Driver Info
5	**API : enterDest** **Type : POST**	User enters the destination address after requesting a ride	**Input:** Geolocation (Latitude and Longitude) **Output:** Status	1. User after requesting a ride selects the destination address on the map

				2. Returns Best route available. 3. Returns Status
6	**API : receiveETA** **Type : GET**	User receives ETA (Estiamated time of arrival) after providing destination address	**Input :** Ride ID or Ride from requested ride list **Output :** ETA	1. User selects a ride from list 2. Returns ETA
7	**API : splitFare** **Type : POST**	After a ride starts, Split Fare allows you to add the other passengers — they'll each pay a small transaction fee, but otherwise all credit cards will be charged equally.	**Input :** Passengers traveling together **Output :** Status	1. User adds passenger 2. Enter the split ratio 3. Returns the Status of passengers payments
8	**API : shareEta** **Type : GET**	Once the driver has picked user up, user can Share ETA	**Input:** Text message	1. User edit stored message 2. Click send to share ETA with friends.

Customer:

Use Case	Function	Description	Inputs/Outputs	Algorithm
1	**API : signIn** **Type : POST**	Helping the returning customers sign-in and used the site.	**Inputs:** Username, Password **Output:** Success/Failure	**4.** User enters username **5.** User enters password **6.** Click login
2	**API : signUp** **Type : POST**	For new users to sign-up for the services.	**Inputs:** User details **Output:** Confirmation Status	**4.** User enters all his necessary personal data **5.** On successful signUp with no redundant username, user created
3	**API : logout** **Type : GET**	For users to logout from the site.	**Inputp:** No input **Output:** Estimate of ride	4. User provides source and destination 5. User selects route preference i.e. Shortest or Economical 6. Outputs the fare estimate
4	**API: profile** **TYPE : GET**	Fetch user profile data from the database and display it	**Input:** Username **Output:** User profile	3. User clicks on the profile tab. 4. Returns user Info
5	**API : rewards** **Type : GET**	Fetch user reward points and reward starts earned for the customer	**Input:** Username **Output:** User rewards	4. User clicks on the rewards tab. 5. Returns user rewards Info
6	**API : rideHistory** **Type : GET**	All ride histories.	**Input:** Username **Output:** User rewards	3. User clicks on the rides tab. 4. Returns user rides history Info

7	**API : settings** **Type : POST**	User profile setting data fetch. Allows user to change settings and preference data	**Input:** Username **Output:** User settings	4. User clicks on the settings tab. 5. Returns user rides history Info
8	**API : paymentDetails** **Type : GET**	Payment history and payment details for the customer.	**Input:** Username **Output:** User settings	3. User clicks on the payments tab. 4. Returns user payment history Info
9	**API : help** **Type : POST**	Help documentation and setup for the user.	**Input:** No Input **Output:** help page	1. User clicks on the help tab.

Drivers:

Use Case	Function	Description	Inputs/Outputs	Procedure
1	**API:** registerNewDriver **Type: POST**	Register new driver	**Inputs:** first name, middle name, last name, licence id and address. **Output:** Confirmation Status	**1.** Driver log in **2.** Registers himself/herself has driver with its basic information.
2	**API :** deleteDriver **Type : POST**	Delete Registered Driver	**Inputs:** Driver name, Car number **Output:** Confirmation Status	**1.** Admin Logs in 2. Choose specific driver's information
3	**API :** getAllDrivers **Type : GET**	Get information of all drivers	**Input: No input** **Output:** Collection of Drive's information	1. Admin logs in 2. Get All driver's information.
4	**API:** updateDriverInfo **TYPE : POST**	Update Driver's information	**Input:**Driver Name, id and address	1. Driver's logs in 2. Update Driver's information.

			Output: Driver Info	
5	**API :** getDriverInfo **Type : POST**	Get Driver's information	**Input:** Driver Name **Output:** Driver's information	1. Admin logs in 2. Get specific drivers information
6	**API :** getVideofDriver **Type : GET**	Get Video of Driver depicting information	**Input :** driver name **Output :** Driver's Video	1. User Logs in 2. Provides drivers name and get video of driver.
7	**API :** acceptPayment **Type : POST**	Accept Payment	**Input :** transaction id **Output :** Status	1. Payment is deducted from user's account. 2. Driver's accept information.
8	**API :** registerCar **Type : GET**	Register Driver's Car	**Input:** Car id	1. Driver's login in 2. Register's its car
9	**API:** updateCarInfo **Type: POST**	Update Car Information	**Input:** Car information	1. Driver's log in 2. Updates car information
10	**API:**updateDriverTimings **Type: POST**	Update Driver's Timings.	**Input:** Time range **Output:** Status	1. Driver's log in 2. Update driver's information
11	**API:** updateDriverArea Type: POST	Update Driver's location	**Input:** Driver's current location Output: Status	System updates Drivers location
12	**API:** updateDriverRating Type: POST	Update Driver's Rating	**Input:** Driver's rating Output: Status	1. User logs in 2. Rates Driver and driver's ratings are updated.

Admin:

Use Case	Function	Description	Inputs/Outputs	Algorithm
1	**API : addDriverToSystem** **Type : POST**	Add the driver to the system.	**Inputs:** Driver Details **Output:** Confirmation Status	**1**.Admin logs in 2.Goes to the driver section. 3.Add the driver details to the system.
2	**API : deleteDriverToSystem** **Type : POST**	Deletes the driver from the system.	**Inputs:** Driver Details. **Output:** Confirmation Status	1. User selects the driver to be deleted 2. Success message on successful deletion and error if deletion failed.
3	**API : addCustomerToSystem** **Type : POST**	Add the customer to the system.	**Input:** Details of customer **Output:** Confirmation Status	**1**.Admin logs in 2.Goes to the customer section. 3.Add the customer details to the system.
4	**API: editDriverInfo** **TYPE : POST**	Edits the Driver information to be edited.	**Input:** Search Driver details to edit. **Output:** Confirmation Status	1.Searches the Driver details. 2. Edits the Driver information.
5	**API : editCustomerInfo** **Type : POST**	Edits the Customer information to be edited.	**Input:** Search Customer details to edit. **Output:** Confirmation Status	1.Searches the Driver details. 2. Edits the Driver information.

7	**API : displayBill** **Type : POST**	Display the Bill with required details of Bill.	**Input:** Search the Bill based on the search criteria. **Output :** Display Bill	1.Search the Bill on the required criteria. 2.Display the Bill based on the search criteria.
8	**API : revenuePerDay** **Type : POST**	Shows the Statistics of the revenue per day.	**Input:** Enter the day to be selected for statistics. **Output :** generate data for statistic.	1.Enter the day on which statistics is required. 2.Generates data required for the statistics .
9	**API : totalRides** **Type : POST**	It Shows the total rides for a particular location.	**Input:** Enter the location on which total rides need to be displayed. **Output :** generate data of the total no of rides.	1.Enter the location on which total rides need to be found. 2.Generates data required for the total rides of a particular location.
10	**API : analyzeRidesPerDay** **Type : POST**	Shows analysis using graph or chart of rides per day.	**Input:** Enter the day for analysis of rides. **Output :** generate data for analysis.	1.Enter the day on which ride analysis is to be done. 2.Generates data required for the analysis.
11	**API : analyzeRidesPerDriver** **Type : POST**	Shows analysis using graph or chart of rides per driver.	**Input:** Enter the driver for analysis of rides per driver. **Output :** generate data for analysis.	1.Enter the driver on which ride analysis is to be done. 2.Generates data required for the analysis.
12	**API : analyzeRidesPerCustomer** **Type : POST**	Shows analysis using graph or chart of rides per customer.	**Input:** Enter the customer for analysis of rides per customer. **Output :** generate data for analysis.	1.Enter the customer on which ride analysis is to be done. 2.Generates data required for the analysis.

Billing:

Use case	Function	Description	Inputs/Outputs	Algorithm
1	API: registerCreditcard Type: POST	Lets the customer scan and link credit card with the Uber account.	Input: Customer credentials Output: credit card linked with the Uber account	1.User scans credit card 2. Enters credit card details 3.register
2	API : registerPaypalAccount Type : POST	Lets the customer link his paypal account with the Uber account.	Input: paypal account credentials output: paypal account linked	1.Enter paypal account details. 2. Register
3	API:registerGoogleWallet Type: POST	Lets the customer link his google credentials with the Uber account.	Input: GoogleWallet credentials Output: GoogleWallet account linked	1.Enter Google Wallet details. 2.Register
4	API: registerApplePay Type:POST	Lets the customer link his Apple Pay account with Uber.	Input:ApplePay account credentials Output: ApplePay account linked with Uber Account	1. Enter Apple ID, password 2. Register
5	API :generateNewBill Type: GET	generates the ride bill(using the dynamic pricing), detailing the fare break,trip statistics and the Driver.	Input: Ride Distance, time and price surge/drop statistics Output: generated bill	1.Ride statistics and automated pricing algorithm applied 2. Bill generates accordingly

6	API :deleteBill Type: DELETE	deletes an existing bill.	Input: Bill ID Output: Bill deleted from customer history.	1.Search the bill to delete by entering any of the search criteria 2.Delete bill history.
7	API :searchBill Type: GET	searches an existing bill using data, time , customer information.	Input: Bill ID, Customer ID, Date, Time, Driver ID Output: fetch the bill	Search the bill by entering any of the attributes - Customer ID, Driver ID, Time, Date.
8	API: payBill Type: POST	lets the customer pay his bill using one of the several payment options provided.	Input: Select the mode of payment Output: Bill payment confirmation	1.select the mode of payment. 2. Pay
9	API : autoAddtip Type: POST	Lets the customer tip the driver, starting 20%	Input: Driver ID, Ride ID, Customer ID Output: Payment credited to Driver ID	1. Select the % to tip 2. Pay Tip
10	API : usePromocode Type: GET	lets the user avail discounts on promo code.	Input: Enter the promo code Output: Avail the discount	1.Enter the code 2.get deduction from the bill
11	API : useFreerides Type: GET	use the free rides(introductory and other offers).	Input: Enter the introductory coupon code Output: deduct the partial amount/ full amount from the bill generated.	1. Enter the code 2. get the amount deducted from the bill generate. 3. Pay the bill
12	API : referCustomer Type: POST	by referring to a new customer, avail discounts on rides.	Input: Enter Customer Name, ID Output: get discount from the bill	1.Enter the customer name, ID 2. check if the customer has used the service.
13	API : paymentHistory Type: GET	lets the customer see his payment history.	Input: Customer ID Output: Payment History	1.Search the customer using Customer ID 2.fetch payment history.