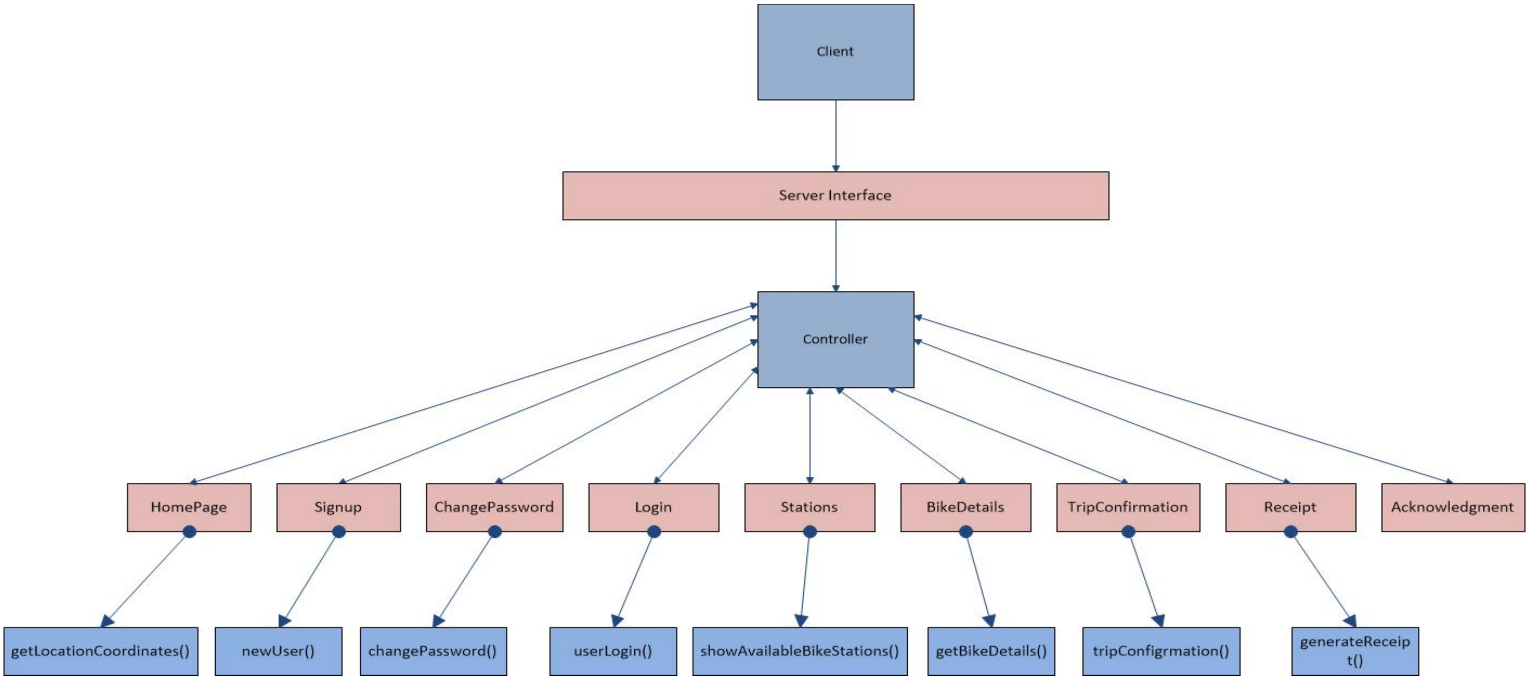
BIKE SHARE
wheels when you want them

downtownbiking.com

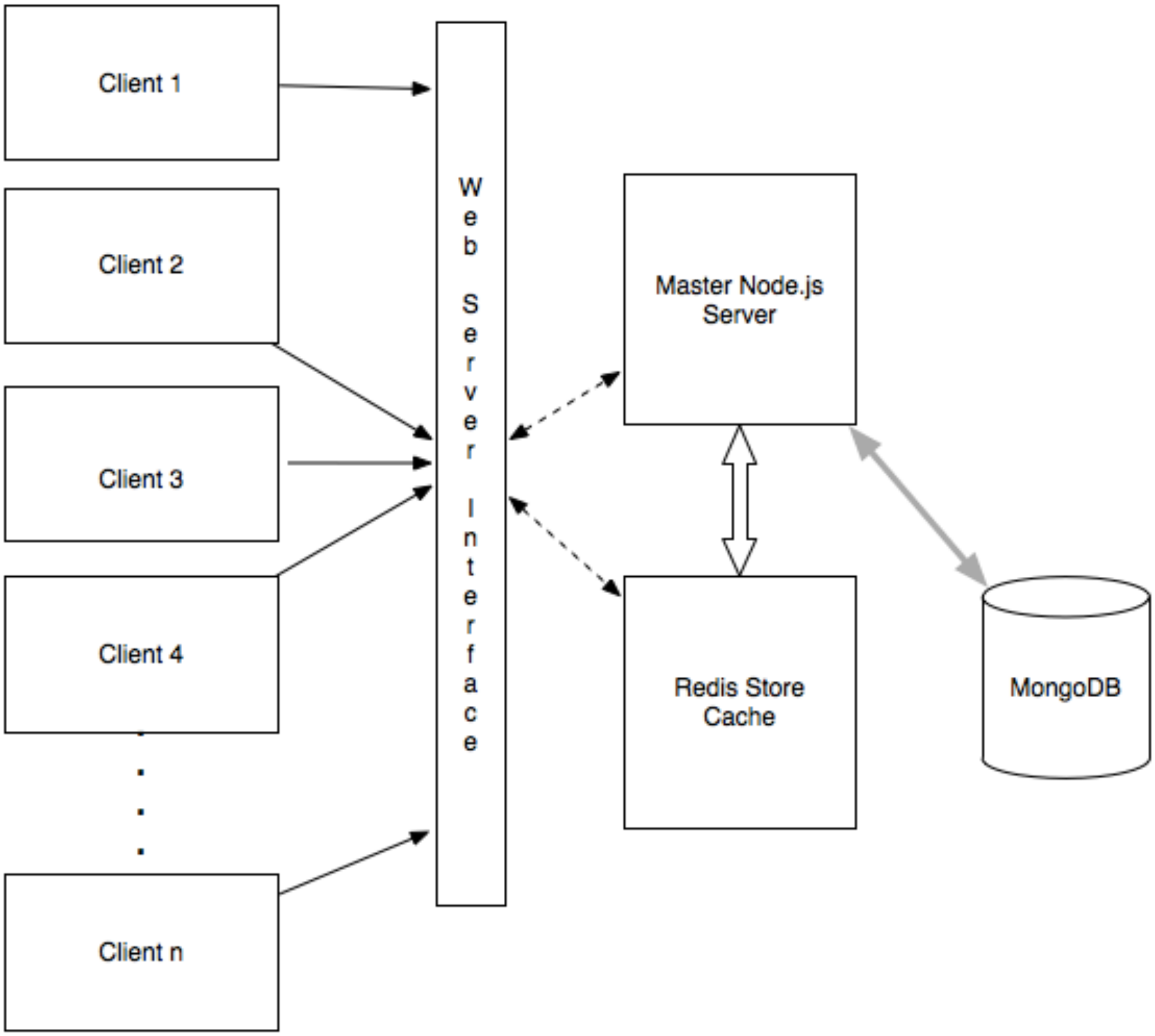
Dataflow Diagram – User Perspective



BIKE SHARE
wheels when you want them

downtownbiking.com

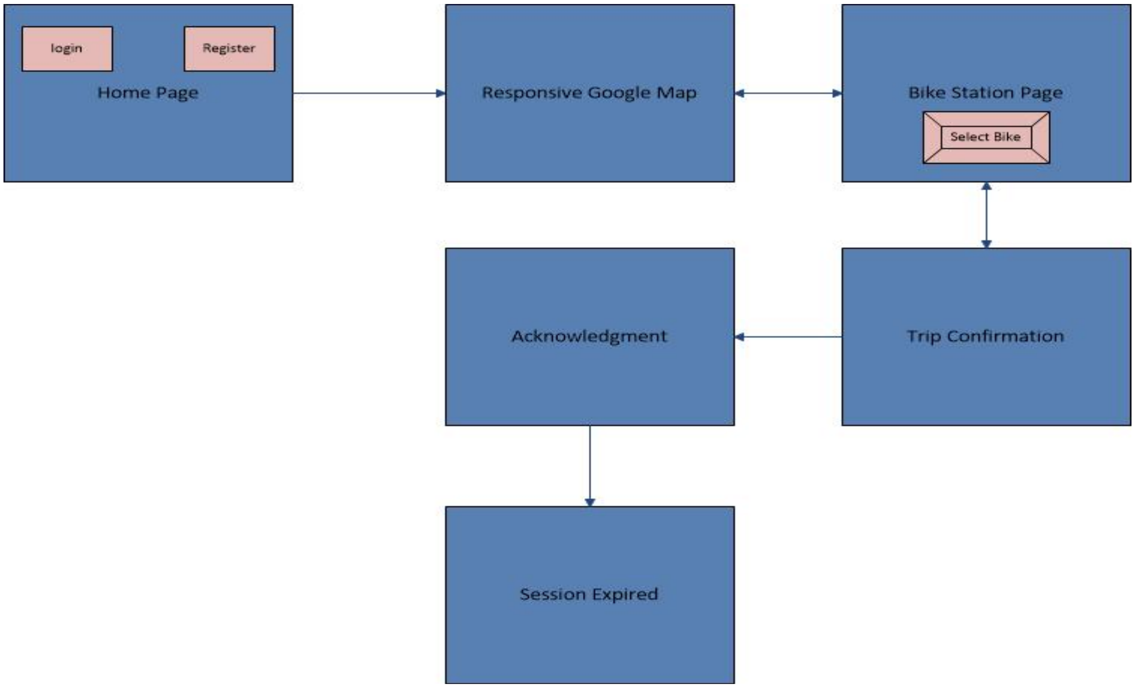
Architecture Diagram



BIKE SHARE
wheels when you want them

downtownbiking.com

User Experience



BIKE SHARE
wheels when you want them

downtownbiking.com

Features Set

No	Feature	Status
#1	Get all bikeStations where bikes are available	Completed
#2	Get all bikeStations where parking slots are available	Completed
#3	Get bike types and available slots for a selected bikeStation	Completed
#4	Get email receipt of trip confirmation along with tripId	Completed
#5	Bike availability status update using bikeId	Completed
#6	Bike cost per hour calculated using category, insurance, time, and location priorities as decorators over the base price per hour	Completed
#7	Extended bike reservation request time	Completed
#8	Pick a dropOffPoint from a list of nearby dropOffLocations from a drop down list at the end of the trip	Completed
#9	Admin function – find all trips, or trips by tripId	Completed
#10	Admin function – find all transactions, or a specific transaction by transactionId	Completed
#11	Admin function – add new bikeStations and bikes	Completed
#12	Admin function – remove bikeStations, bikes, trip details, transactions	Completed
#13	Admin function – Update the decorator percentages in terms of locationPriority of a station, categoryPrioriy	Completed
#14	Rent bikes from nearby distributed sources – personal bikes	Pending
#15	Return bikes back to the distributed source – personal bikes	Pending

Future Enhancements

#1	Integrate Payment Gateway – Braintree/Paypal
#2	Implement UserPassword Encryption
#3	Implement Application Firewall
#4	Implement MongoDB replica sets on remote instance
#5	Improve code practices using connection pooling and caching
#6	Deploy Client Side on downtownbiking.com
#7	Deploy Server Clusters on AWS EC2

Technology Stack

Node.js | Express | MongoDB | Redis | Javascript | JQuery

Reasons for choosing technology

Web Server: Node.js provides asynchronous non-blocking I/O event loop programming paradigm to handle concurrent HTTP requests at scale. Clustering of such Node servers provides added load balancing infrastructure and functionality, to process requests.

Database: MongoDBis a NoSQL document type database, which has good support when coupled with Node.js, via npm installable node module. Being a document type database, JSON objects can be dumped into MongoDB, without the need for any further conversions. Moreover, the NoSQL database can scale while performing very large number of simultaneous reads/writes, by offering its built-in concurrency control mechanisms.

JavaScript: Maintains uniformity of technology stack at client/server end points. It can be leveraged to create AJAX requests.