

**WUHAN UNIVERSITY**

**Project Name: Ride sharing service**

**Subject Name: Modern SoftwareArchitecture**

**Teacher: Liang Peng**

**Group Members**

**Name: MIA MD SHAMIM**

**ID: 2022272110001**

**Name: MD MONIR HOSEN JONY**

**ID: 2022272110005**

**Name: AKTAR MST SHAMIMA**

**ID: 2022272110008**

**Project Name: Ride sharing service**

**Project description:** When it comes to getting about in cities, many people choose ride-sharing apps like Uber or Lyft. Indeed, such apps make short-range commute very easy by offering competitive pricing, reasonably short wait time, high security and high availability. From a technical point of view, systems like these are very interesting because nearest-neighbor searching is hard. These systems have much more complex architecture and there are a lot of components joined together internally to provide riding services all over the world.

A user can request a ride through the application and within a few minutes, a driver arrives nearby his/her location to take them to their destination. It has a backend service, a frontend service, and a single database.

1. **Business goals:**
2. To achieve 99.99% reliability of the core travel experience on ride sharing service (only have a total of one hour of downtime per year, and a maximum of one minute per week, in other words, every 10,000 operations can only fail 1 time at a time.
3. Codebase divided by 2: Core code, and optional code. Core code is compulsory when the rider register, calls and completes or cancels travel requirements. Any modification to the core code must go through a rigorous review process. Optional code is less reviewed and can be dynamically closed at any time. This encourages mutual independence at the code level, allowing us to try new features and stop them at any time.
4. Core architecture: class names, inheritance relationships between business logic units (inheritance relationship between business logic units), main business logic, plugin point (name, dependency, structure, etc.), reactive programming chains (relationship between reactive programming), unified platform components (unified platform-level modules).
5. **stakeholders and users**

The main stakeholders and users of the ride sharing service are the admin (ride sharing company), Passengers and Drivers.

1. **Requirements**
   1. Riders can view nearby drivers
   2. Riders can request a ride
   3. Riders can view the driver’s ETA and approximate price
   4. Driver accepts the trip; the rider can view the driver’s location and communicate until trip completion.
   5. Book a cab
   6. Match riders with drivers
   7. See cabs in your vicinity
   8. Location tracking
   9. Post-tractions: Collect ratings, send emails, update databases, schedule payments
   10. Price and Surge: the price is increased when there are more demand and less supply with the help of prediction algorithms. According the ride-sharing service helps to meet supply demands by increasing the price, more cabs will be on the road when the demand is more.
2. **Quality Attribute:**

**Consistency:** Mission-critical applications such as financial dashboards require data to be consistent across all regions. This includes zero data loss in the inter-region and intraregional dispersal and processing mechanisms, de-duplication as well as ability to certify data quality.

**Availability:** The real time data infrastructure stack must be highly available with 99.99 percentile guarantee. Loss of availability has a direct impact on ride sharing business and may result in significant financial losses. For instance, dynamic pricing leverages the real-time data infrastructure component for calculating demand and supply ratios per geo-fence, which in turn is used to influence the price of a trip.

**Scalability:** The raw data streams constitute petabytes of data volume collected per day across all regions. This data is constantly growing based on organic growth of our user base, new lines of business deployed by ride sharing company as well as new real time analytics use cases that arise over time. The ability to scale with this ever-growing data set in a seamless manner, without requiring users to re-architect the processing pipelines is a fundamental requirement of the real-time data infrastructure stack.

**Flexibility:** We need to provide programmatic as well as declarative (SQL like) interface for expressing computational logic to accommodate the diverse user groups. In addition, some use cases need a push-based model which is semi stateful and continuously emits generated results whereas others might need a stateful pull-based model where the user can execute queries on the raw data stream. For instance, users can create intelligent alerts in case of business rule violation using push-based stream processing pipelines. Whereas, dashboarding and triaging will require a pull-based SQL interface for the same datasets.

**Security:** This system has multistep security. The user needs to login to the system every time by providing a password and user ID, and the login session will time out after every 2 minutes to ensure high security. When the passenger pays the bill, they need to again give the password. All of those steps will keep their bank card and payment information secure. In bellow see full description about security.

Basic security behaviors:

* Authentication: Login using at least a user name and a password
* Authorization: according to their profile, online user must be granted or not allowed to receive some specific services (Automatic match finding, Ride Suggestion, etc...

For internet access, the following requirements are mandatory

* Confidentiality: sensitive data must be encrypted if any (credit card payments).
* Safety: Credit card data must not be kept at a local database.
* Data integrity: Data sent across the network cannot be modified by a tier
* Auditing: Every sensitive action can be logged
* Non-repudiation: gives evidence a specific action occurred

**Performance:** This system has high performance ability. Passengers can book a taxi and search for availability in a short period of time. Admin can update the bike taxi list. Search queries should return 90% of the time below 5 seconds. The credit card payment transaction should finish in 10 seconds.

1. **Viewpoint**

The system has some viewpoint.

* Use-Case view
* Logical view
* Development view

**GitHub Link:** https://github.com/shamimaaktar1/SA-Project-Ride-Sharing-Service