# CS 255 System Design Document Template

This template lays out all the different sections that you need to complete for Project Two. Each section has guidance to prompt your thinking. You will need to continually reference the interview transcript as you work to make sure that you are addressing your client’s needs. There is no required length for the final document. Instead the goal is to complete each section based on what your client’s needs are. Remove this note when you are finished, and replace all bracketed text with the relevant information.

## UML Diagrams

### UML Use Case Diagram

A diagram of a driver pass

Description automatically generated

### UML Activity Diagrams

*A diagram of a process

Description automatically generated*A diagram of a customer

Description automatically generated

### UML Sequence Diagram

A diagram of a customer

Description automatically generated

### UML Class Diagram

A diagram of a software company

Description automatically generated

## Technical Requirements

**Hardware Requirements**

To make sure the system runs smoothly for both customers and drivers, we need hardware that can handle multiple people using it at the same time.

* Servers: We'll use cloud servers like AWS or Google Cloud. These are good because they can grow as more people use the app. The server should have:
  + At least 16 GB of RAM
  + A 4-core processor (vCPU)
  + 500 GB SSD for storing user and ride data
* User Devices: People will use smartphones or computers to access the system. So, it should work well on:
  + Phones with iOS 14+ or Android 10+
  + PCs or laptops with at least 4 GB of RAM and good internet
* Internet/Network:
  + Users need a stable connection, at least 10 Mbps
  + Servers should be hosted in places with fast internet and backup options in case something goes down

**Software Requirements**

The system includes apps for users and a backend to process data and store it.

* Frontend (User App):
  + Can be built using Flutter or React Native for mobile apps (works on both iOS and Android)
  + Or, make a web app using React.js or Angular for people using browsers
* Backend (Server Side):
  + Can use Node.js with Express or Django (Python) to handle requests and logic
  + Will process actions like ride requests, user sign-ups, and driver updates
* Database:
  + Use PostgreSQL or MySQL to store structured data like user info and ride history
  + Use MongoDB for more flexible data like real-time updates or logs
* Operating System:
  + Run servers on a reliable Linux OS like Ubuntu 20.04 LTS

**Tools and Frameworks**

These are the tools we’ll use to build, test, and maintain the app.

* Development Tools:
  + Use VS Code or IntelliJ IDEA to write the code
  + Use Git and host the code on GitHub or GitLab to keep track of changes
* Testing Tools:
  + Use Jest (for JavaScript) or PyTest (for Python) for testing the code
  + Use Selenium or Appium to test the user app automatically
* DevOps Tools:
  + Use GitHub Actions or Jenkins for automating testing and deployment
  + Use Docker to package the app so it works the same everywhere
* Monitoring & Analytics:
  + Use Prometheus and Grafana to check server health
  + Use Google Analytics or Mixpanel to understand how users interact with the app

**Infrastructure Requirements**

The system must be secure, reliable, and able to grow as more people use it.

* Cloud Hosting:
  + Host everything on AWS, Google Cloud, or Azure to handle more traffic and provide backups
* Security:
  + Use HTTPS to secure data
  + Use OAuth 2.0 or JWT for logging in safely
  + Encrypt sensitive data like payment info
  + Use WAFs and do regular security checks
* APIs & Integrations:
  + Use Google Maps API for tracking locations and routes
  + Use payment services like Stripe or PayPal for handling payments securely
* Load Balancing:
  + Use tools like AWS Load Balancer to spread traffic across servers so nothing crashes during busy times