# 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 picture containing diagram, text, plan, line

Description automatically generated

### UML Activity Diagrams

A picture containing text, screenshot, diagram, design

Description automatically generated

A picture containing text, screenshot, font, design

Description automatically generated

### UML Sequence Diagram

*A diagram of a customer service

Description automatically generated with low confidence*

### UML Class Diagram

*A diagram of a uml class

Description automatically generated with medium confidence*

## Technical Requirements

1. Hardware Requirments:
   1. Server Hardware: A powerful server is required to host the system application and databases. The details would be determined by the predicted load and number of users. A cloud-based system that is scalable may be great.
   2. Client Hardware: End-users will require a computer or a mobile device with internet connectivity to access the system.
2. Software Requirements
   1. Depending on the technological stack, the server operating system will be Linux or Windows Server.
   2. Database Management System (DMS): SQL-based systems such as MySQL or PostgreSQL for relational data storage, or NoSQL options such as MongoDB for a more flexible data format.
   3. Backend Language/Framework: Depending on team abilities and preferences, Java/Spring Boot, Python/Django, or Node.js/Express.js.
   4. For creating a contemporary, interactive client-side application, use Angular, React, or Vue.js as the frontend framework.
   5. If a mobile application is required, it may be built with Flutter, React Native, or Swift/Kotlin for native apps.
   6. API Documentation and Testing: Postman and Swagger are tools for developing, testing, and documenting APIs.
   7. Git for source code management, together with a platform such as GitHub or GitLab.
   8. Continuous Integration/Continuous Deployment (CI/CD): Automate build, test, and deployment procedures with Jenkins, GitLab CI, or GitHub Actions.
3. Infrastructure
   1. AWS, Google Cloud, or Azure for application and database server cloud hosting. Cloud solutions are both scalable and dependable.
   2. Secure network infrastructure is required for system access. If necessary, use VPNs for remote access.
4. Security:
   1. SSL Certificates: These are used to enable HTTPS and ensure secure data transfer.
   2. Firewall: To protect servers from unauthorized access.
   3. Data encryption is used to protect sensitive data during storage and transmission.
5. Analytics:
   1. Application Performance Monitoring (APM) Tools: Tools for monitoring system performance, such as New Relic or Dynatrace.
   2. Log management solutions include the ELK stack (Elasticsearch, Logstash, Kibana) and Splunk.