Trial Class Application For Maths

Student Teacher communication and learning platform

System and Goal

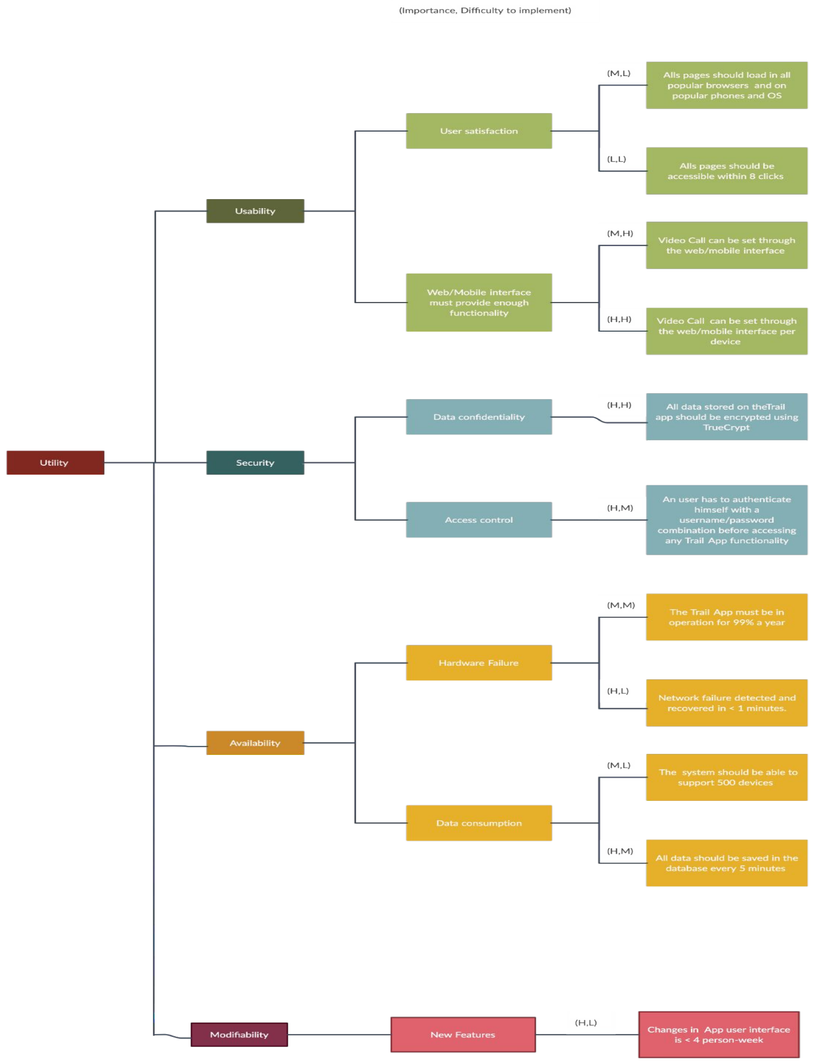
- System allows student (or parent on behalf of student) to book a trail class for Maths education and then have a video class on scheduled time.

- Goal of the system is to have an easy interface to book class and then have a trail class at scheduled time.

Key Requirements

- Users (Student and Teacher) should be able to login in to system (web + mobile)
- Student should be able to book a class through a scheduler
- Student should be able to join class at designated class time
- Teacher should be able to join class at designated time
- Teacher should be able to present an applet related to course for discussion
- Teacher should be able to present an a white board to discuss and draw things
- Student should be able to see Applet and whiteboard
- Applet and whiteboard communication must be two way
- Teacher should be able to send rewards to student in terms of Hats-Off animations and emojis
- Teacher should be able to end class

Utility Tree Diagram


(Importance, Difficulty to implement)
Utility
Usability
User satisfaction
(M,L)
Alls pages should load in all popular browsers and on popular phones and OS
(L,L)
Alls pages should be accessible within 8 clicks
Web/Mobile interface must provide enough functionality
(M,H)
Video Call can be set through the web/mobile interface
(H,H)
Video Call can be set through the web/mobile interface per device
Security
Data confidentiality
(H,H)
All data stored on theTrail app should be encrypted using TrueCrypt
Access control
(H,M)
An user has to authenticate himself with a username/password combination before accessing any Trail App functionality
Availability
Hardware Failure
(M,M)
The Trail App must be in operation for 99% a year
(H,L)
Network failure detected and recovered in < 1 minutes.
Data consumption
(M,L)
The system should be able to support 500 devices
(H,M)
All data should be saved in the database every 5 minutes
Modifiability
New Features
(H,L)
Changes in App user interface is < 4 person-week

Tactics used to achieve ASRs

Usability:

- User Satisfaction and App provided:
    - Made the system browser agnostic
    - Made the system OS agnostic
    - Used Clean Architecture Patterns like onion architecture to create web app and mobile app

Security:

- Data Confidentiality and Access Control :
    - Encryption used at every place
    - Proper auth based system to access any module

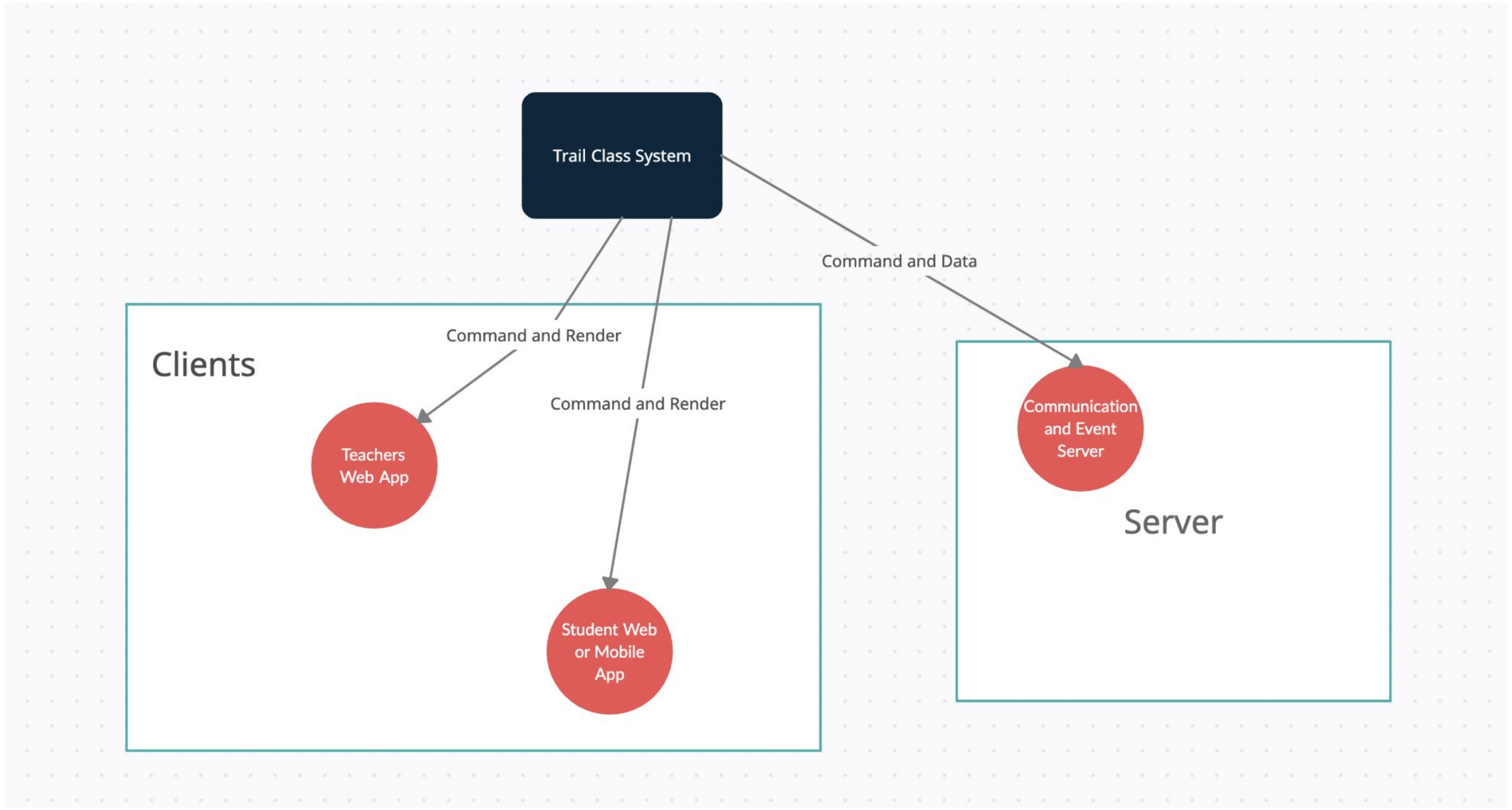
Availability and Performance:

- AWS Cloud and its horizontal scaling system used for 99% available system
- At a time 500 classes can happen parallelly with server
- Autosave of data made possible

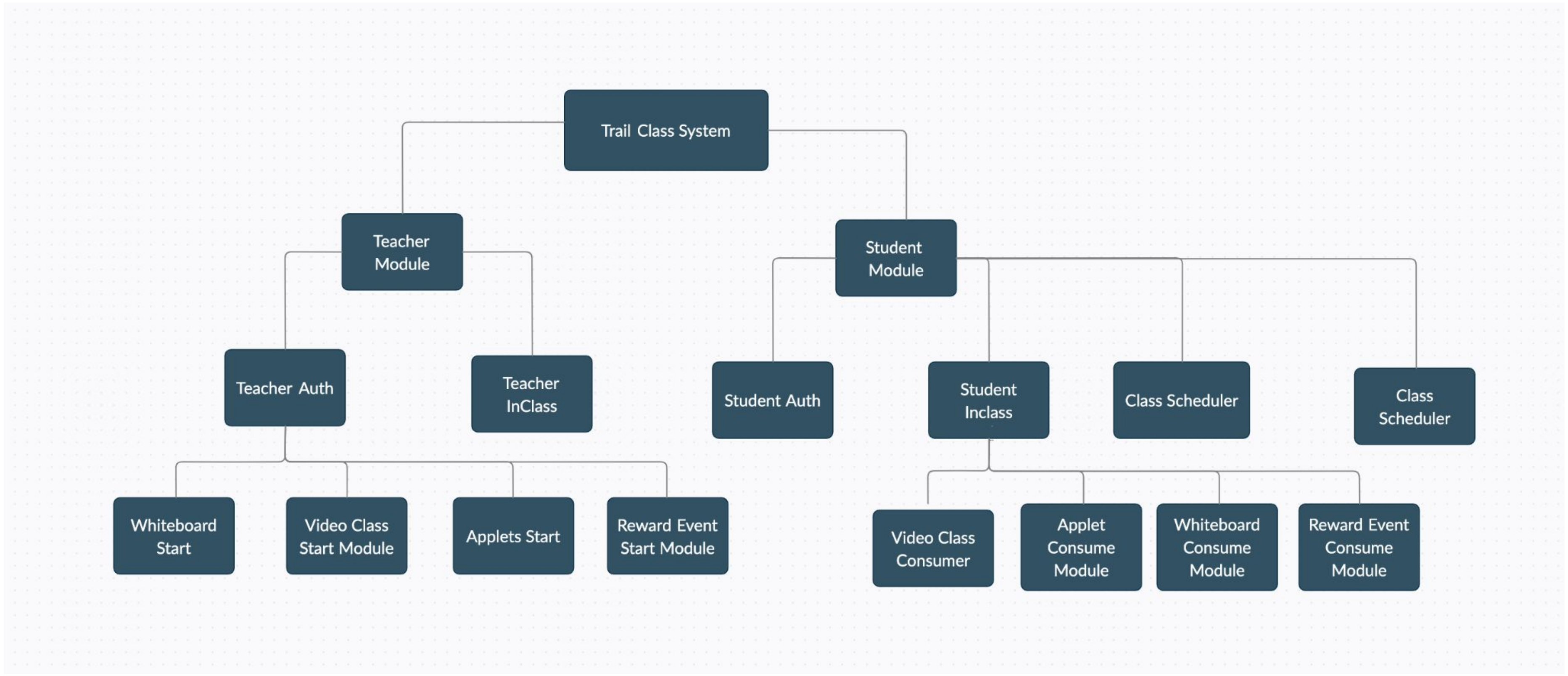
Modifiability:

- Used Clean Architecture Patterns like onion architecture to create web app and mobile app which can be modified easily
- Hyper separation of concern used to achieve this through onion architecture

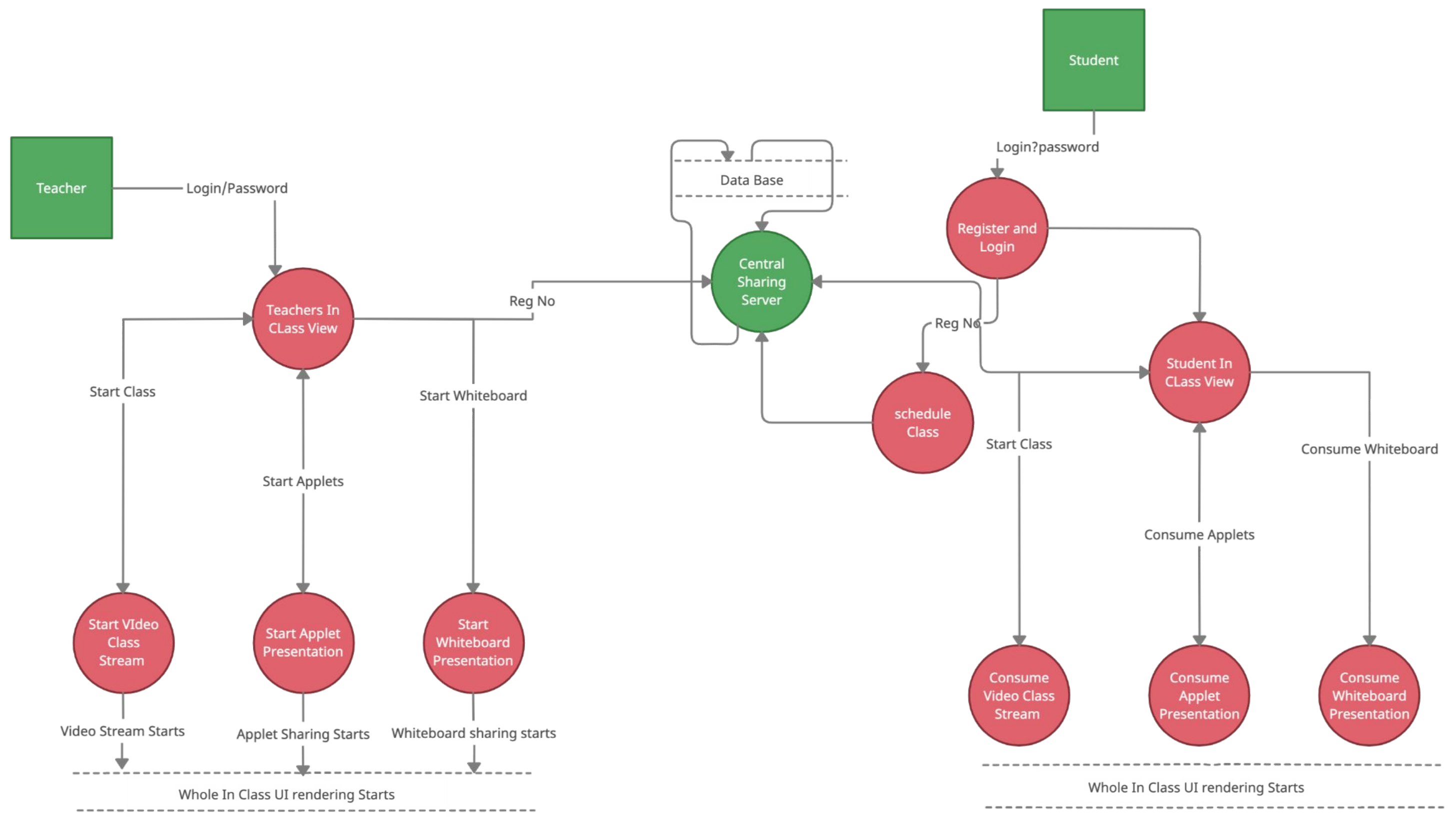
Context Diagram



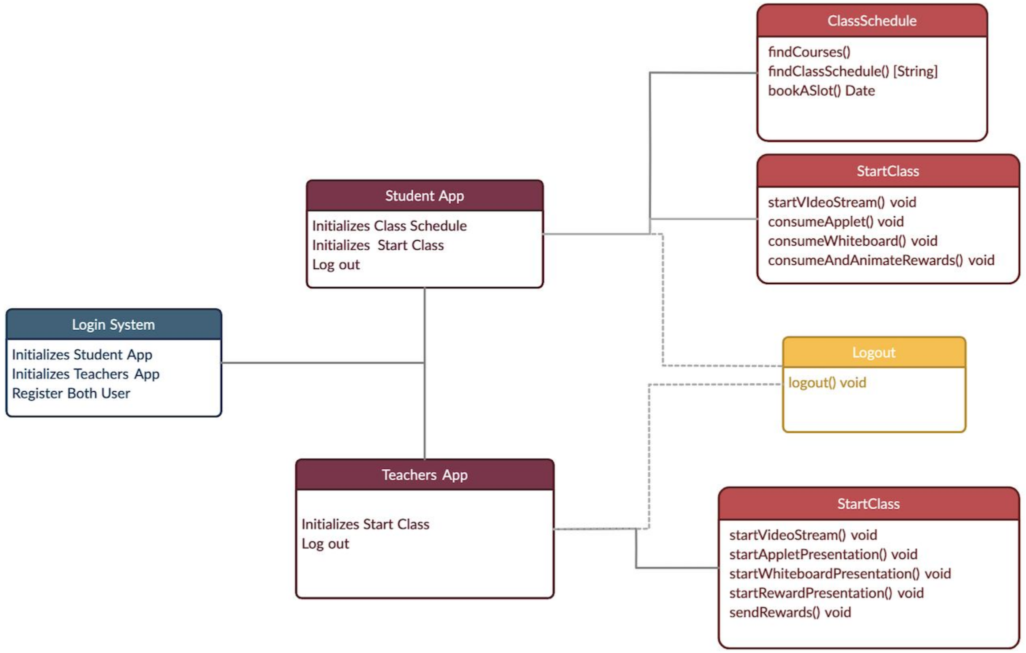
Module Decomposition



Component and Connection Diagram



Deployment Diagram



How System Works

- System is working as Client Server Architecture
- There are two type of Clients which communicate with a central server (Student and Teacher)
- All sort of student and teacher communication happens through central server
- Video/Audio Stream happened with a socket implemented on server
- Both clients have implementation of maths applets and whiteboard which can be shared through a central server.
- Communication between Client and Server happens with REST services and Socket connection
- For the reward events server sends real time events for client to capture
- Web app is created with web technologies like React Js and Mobile Apps are created with React Native
- Server follows the Monolith Service architecture to provide REST APIs and Socket Connections

Key Learning

- Such live video stream system require must plan for ASR and Utility tree and NFRs are as important as FRs

- System is live for an hour so error handlings (Network, Hardware issues) must be at its best and well designed

- User Interface is as important as functionality of the system as different type of users (Teacher/Student) using a complex UI

- System availability and performance is most important concern and must be handled properly

- As system is quite big module decomposition should be planned seriously

- Component connections can not be easily modifiable so must be planned before hand in architecture

- Deployment Diagram should be exhaustive and detailed.