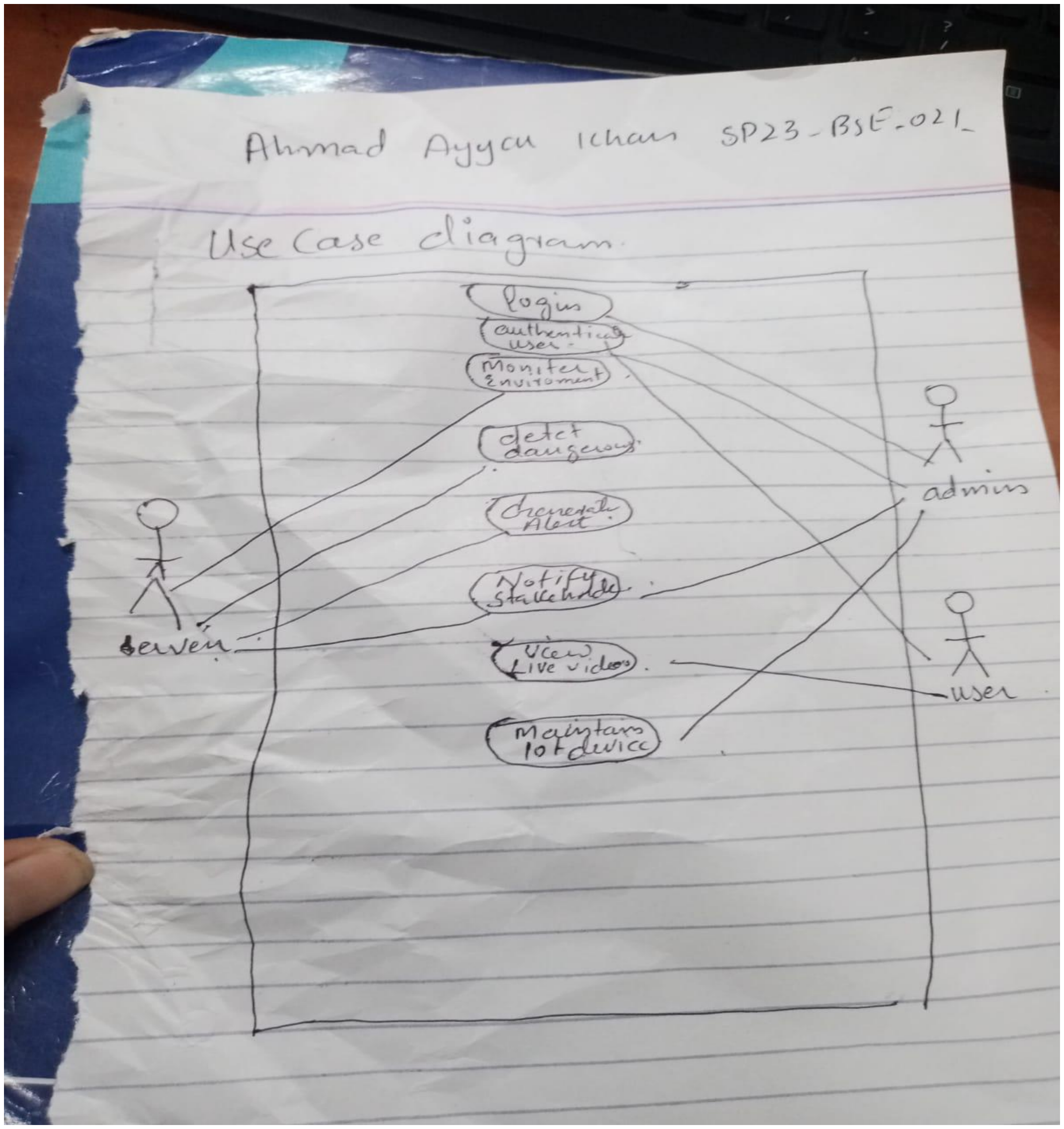
Submitted by :

Ahmad Ayyar Khan

Registration no :

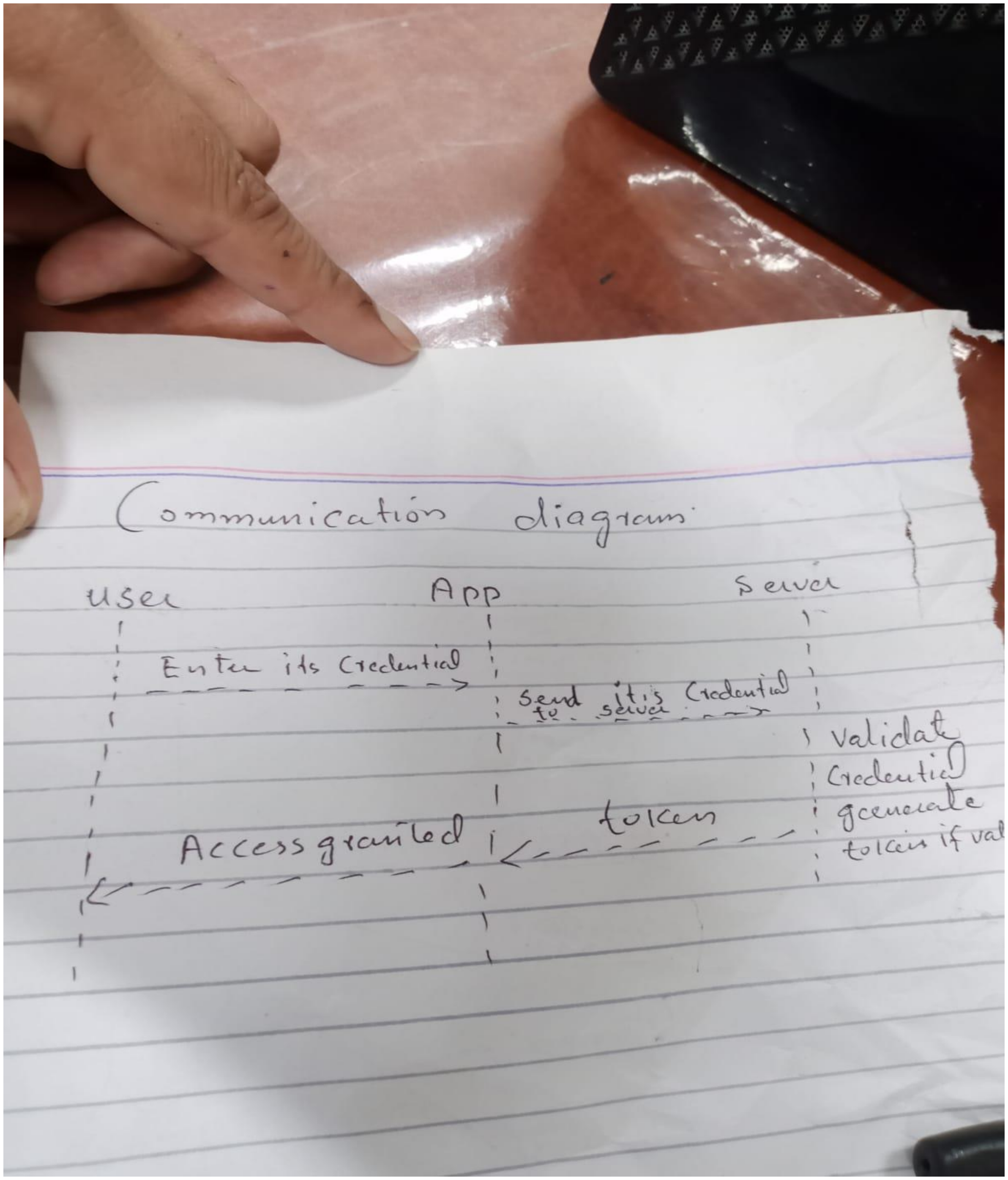
SP23-BSE-021



Ahmad Ayyan khan SP23-BSE-021

Use Case diagram.

Communication diagram


User
App
Server
Enter its Credential
Send itis Credential to server
Validate Credential generate token if val
token
Access granted

SEPARATION OF CONCERN:

WE USE THIS IS PATTERN DUE FOLLOWING BEST APPROACHES WHICH ARE FOLLOWING :

Why This Architecture Principle is best approach for this usecase:

Security

- Usernames and passwords are kept safe because they are checked only on the server.
- Tokens (like JWT) stop the need to send passwords every time, so there is less chance they get stolen.
- Tokens can expire or be cancelled, so if someone steals them, they can’t use them forever.

Scalability

- The client (app) and server work separately, so both can handle many users without slowing down.
- The server can manage lots of login requests at once without making the app slow.

Maintainability & Separation of Concerns

- The app focuses on showing the screen and user interaction.
- The server focuses on checking logins and security.
- If rules about passwords or extra security change, only the server needs to be updated, not the app.

User Experience

- Tokens keep users logged in smoothly without asking for a password every time.
- It supports easy logins like Single Sign-On (SSO) or logging in with Google/Facebook.

Flexibility

- The server can add extra security features like two-step verification or locking accounts easily.
- The same system works for apps on phones, computers, or tablets.

Statelessness and Performance

- Using tokens means the server doesn’t have to remember every user’s login session.
- This makes the system faster and uses less memory