Get Social with

SOC HUB

Empowering College **Societies** &
Elevating **Events** with Ease.

PROBLEM STATEMENT

In India there are around

1070 Universities

this approximately leads to more than

42,000 Colleges

and each college has around 20+ societies, making it around

1 Million College Societies

*Data as of 2020

Managing a society is a hectic task and not only involves team management but getting funds, promoting events, and a hell lot of tedious tasks.

Don't worry **SocHub** comes to the rescue. The only partner you need to run your society smoothly.

What does SocHub Offer?

To Societies

- Efficient Team Management Dashboard
- Robust Event promotion
- Makes Fund Raising a cup of tea.

To Student

- One Stop solution for surfing Events & Competitions.
- Easy Registration.
- Efficient schedule tracking.

ADVANTAGES

- **TARGETED AUDIENCE -** SocHub is a college-specific society management app designed for college students and their societies.
- It offers targeted features and advantages for college social events.
- **TOOLS FOR SOCIETY MANAGEMENT -** SocHub provides society management tools for organizing events, managing members, and tracking finances.
- **MULTIPLE COMMUNICATION CHANNELS -** The app offers multiple communication channels, including group chats, private messaging, and announcements.
- It has event promotion features such as event pages, RSVPs, and reminders.
- **FUNDRAISING -** SocHub allows societies to connect with sponsors directly for fundraising.
- Compared to existing solutions:
  - UnStop: SocHub has a more intuitive user interface, additional features like report generation and fundraising, and eliminates the learning curve.
  - Instagram and LinkedIn: SocHub is college-specific, avoiding distractions and offering a focused platform.
  - WhatsApp: SocHub provides dedicated features for societies, avoiding event promotion spamming and enabling efficient society management.
- Overall, SocHub offers a comprehensive solution tailored to college societies' needs.

FINAL REQUIREMENTS

Authentication System:

- Sign Up Page
- Log in Page
- Google Authentication

Home Screen:

- Event Library (Market Place)
- Home
- Videos
- Polls
- Posts
- Registration

User Profile:

- Details
- Past Participation
- Badges
- Achievements

Explore:

- Events
- Societies

Society Management Dashboard:

- Team
- Meet
- Deadlines
- Roadmap (Tentative)
- Get Merchandise
- Report Generation

Society - Sponsor service (Fundraising Model)

Mutual Marketing

MoU

FRONTEND

Technologies used:

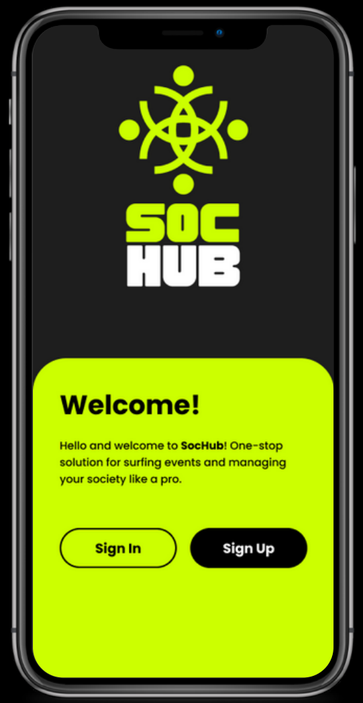
- Flutter

Prototyped using:

- Figma

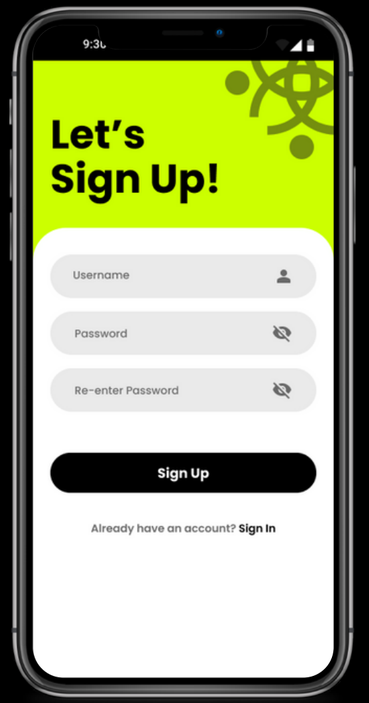
Application User Interface:

- Welcome Page
- Sign Up page
- Sign In Page
- Home - User
- Society Dashboard
- Explore Page
- Profile Page



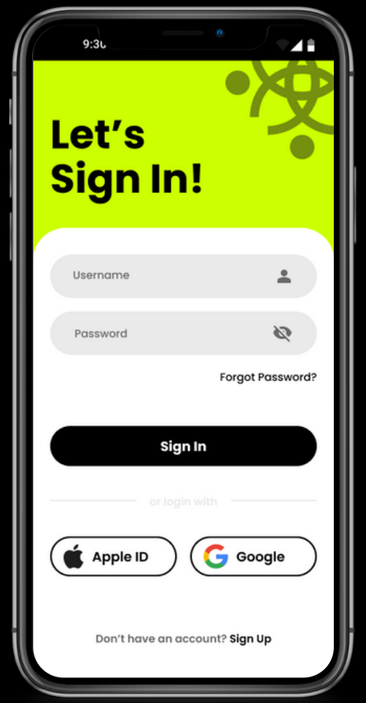
Welcome Screen

User can select either to sign up or sign in.



Sign-Up Page

User can register in SocHUB with E-Mail

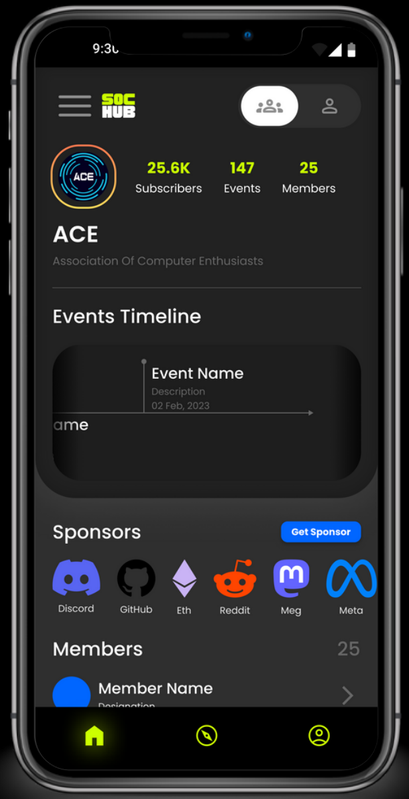


Sign-In Page

User can sign un using email or Google/Apple ID

Home -User

Browse through exciting events posted and register in ones which excite you.



Dashboard - Society

Track your event history, connect with sponsors and keep a check on members.

Explore

Find your favourite societies and follow them to stay updated.

User Profile

Earn badges/rewards for events attended and won.

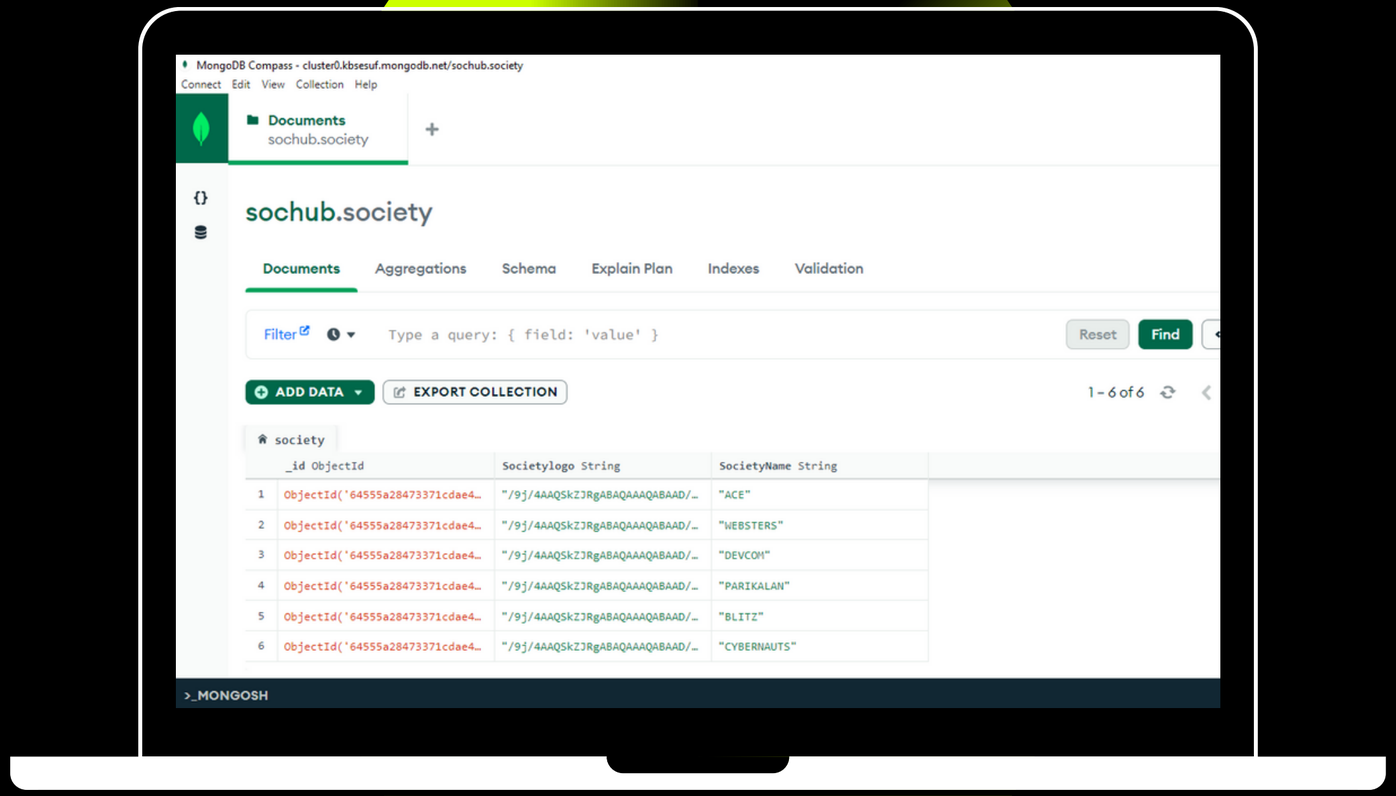
BACKEND

Technologies Used:

- MongoBD

Society Collection:

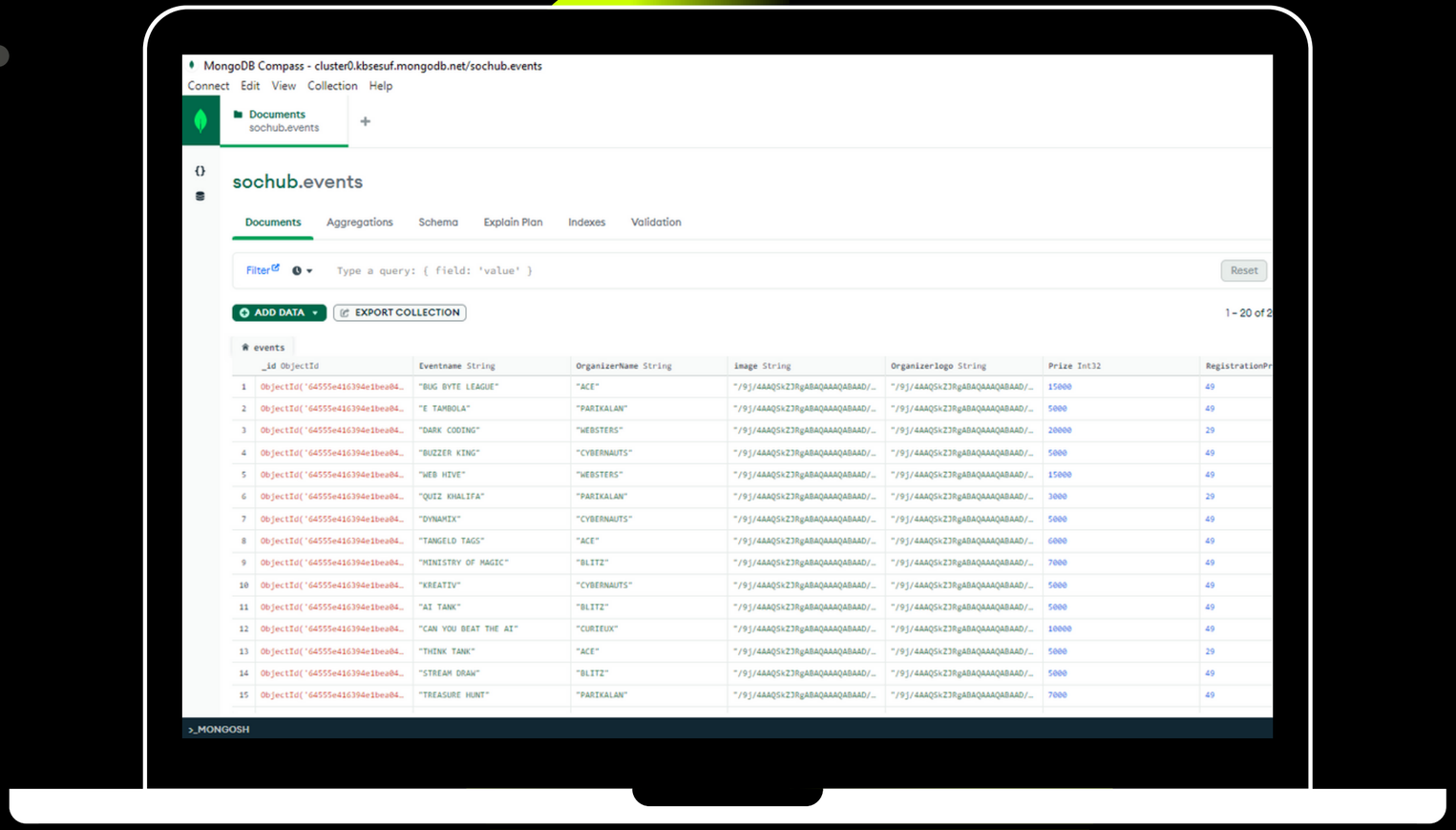
- Variables
  - _id
  - SocietyLogo
  - SocietyName



BACKEND

Events Collection:

- Variables
  - _id
  - EventName
  - OrganizerName
  - image
  - OrganizerLogo
  - Prize
  - RegistrationPrice

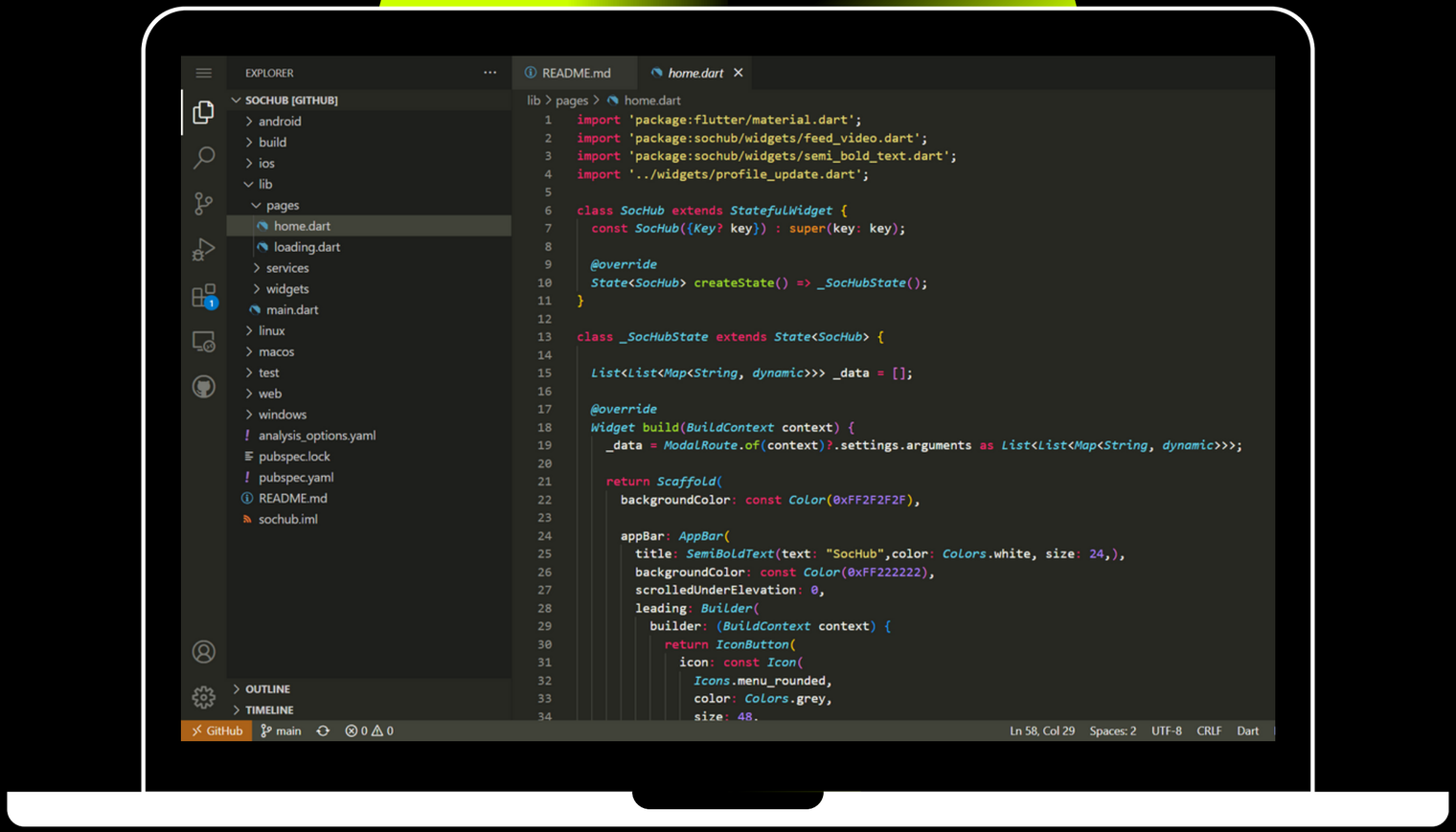


SOURCE CODE

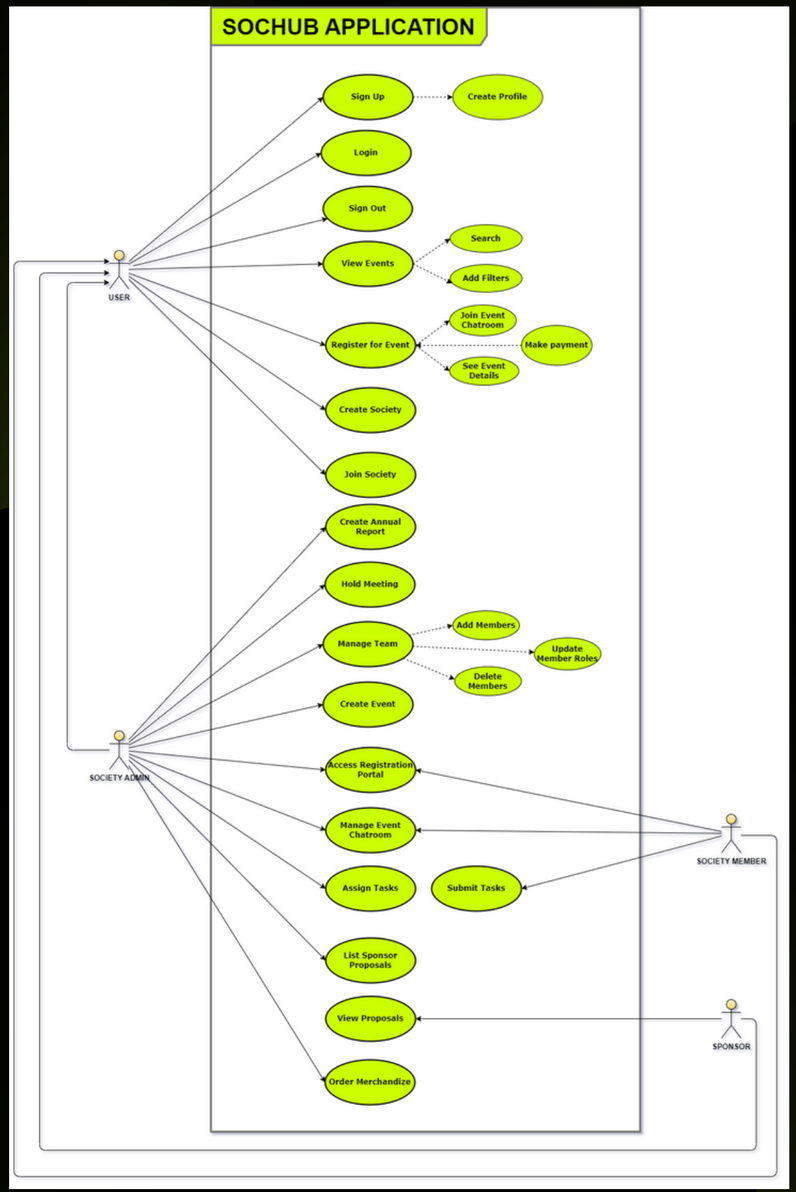
Visit below link for Source Code

github.com/AumGupta/sochub

or scan the QR Code for complete source code of the Application.

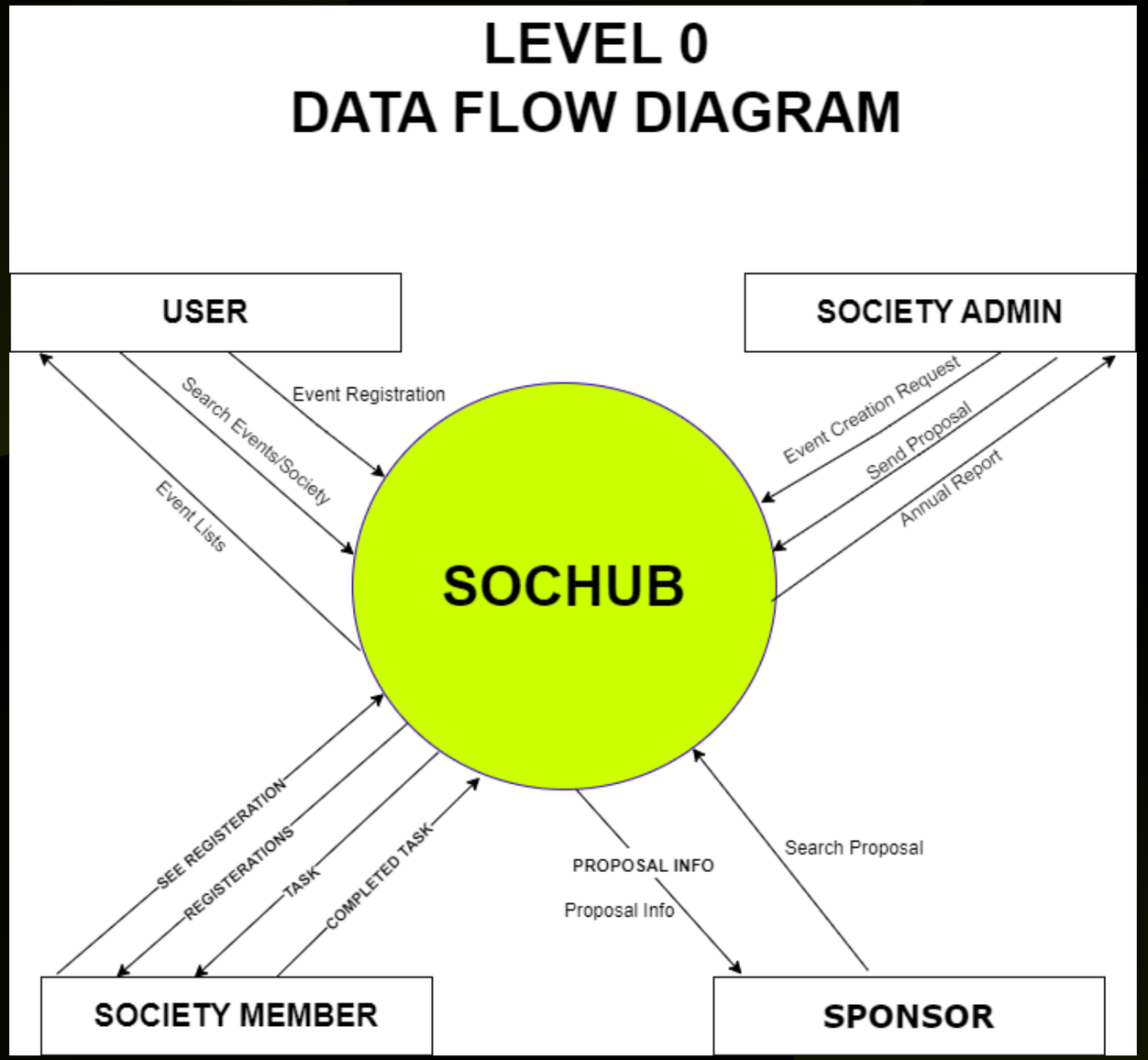


USE CASE DIAGRAM



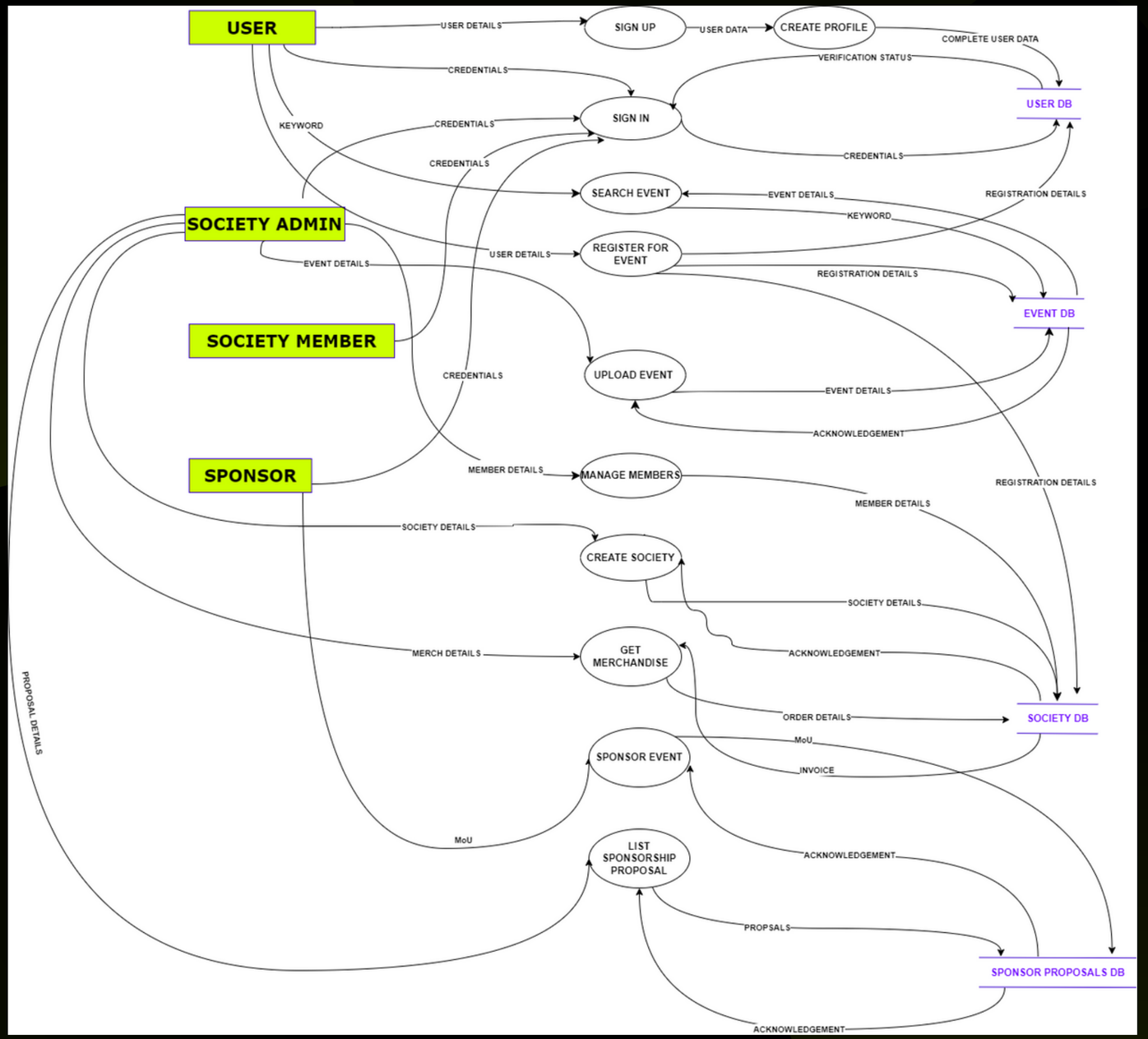
DFD

LEVEL - 0


LEVEL 0
DATA FLOW DIAGRAM
USER
SOCIETY ADMIN
Event Registration
Search Events/Society
Event Lists
Event Creation Request
Send Proposal
Annual Report
SOCHUB
SEE REGISTERATION
REGISTERATIONS
TASK
COMPLETED TASK
PROPOSAL INFO
Proposal Info
Search Proposal
SOCIETY MEMBER
SPONSOR

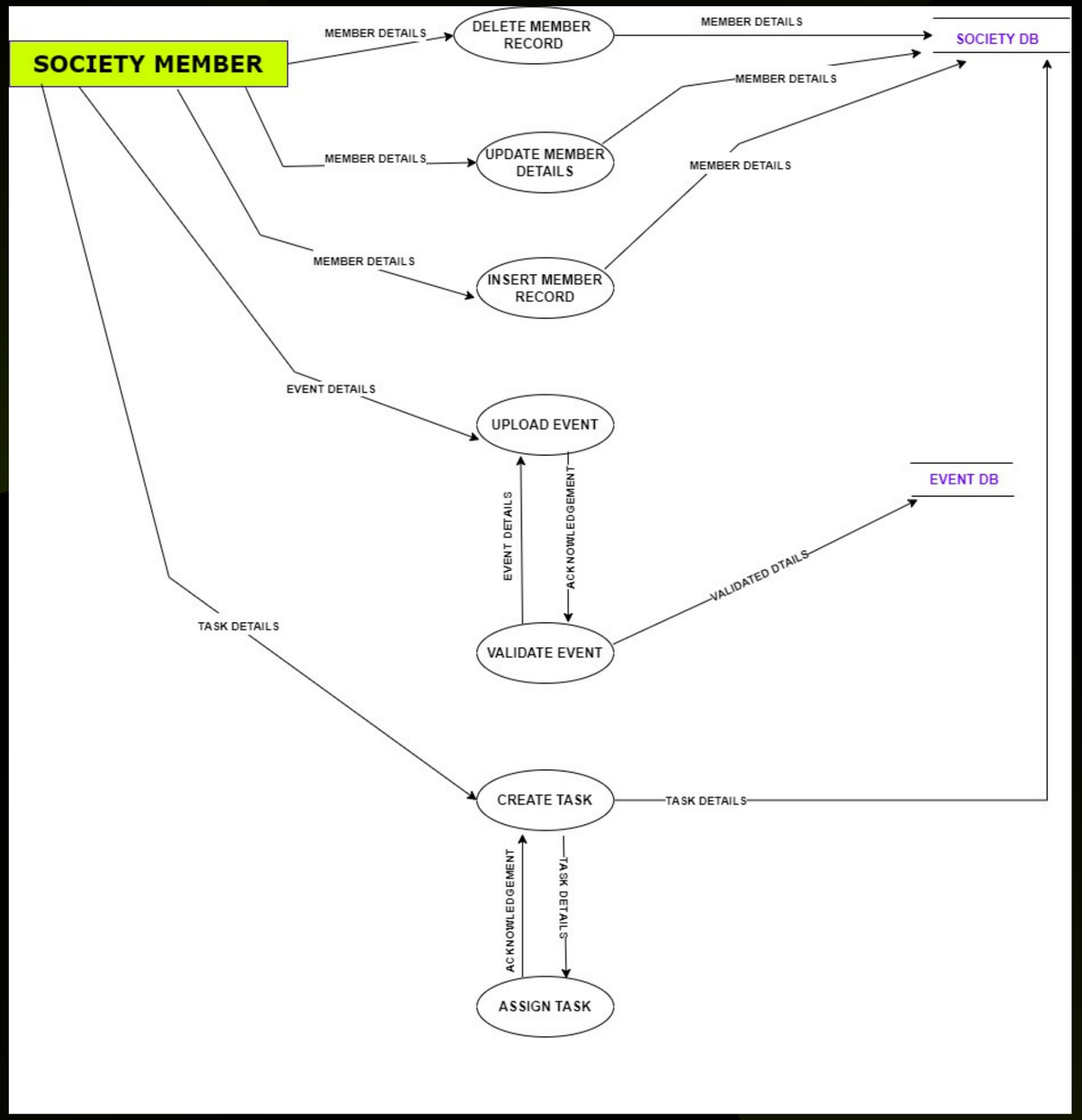
DFD

LEVEL - 1

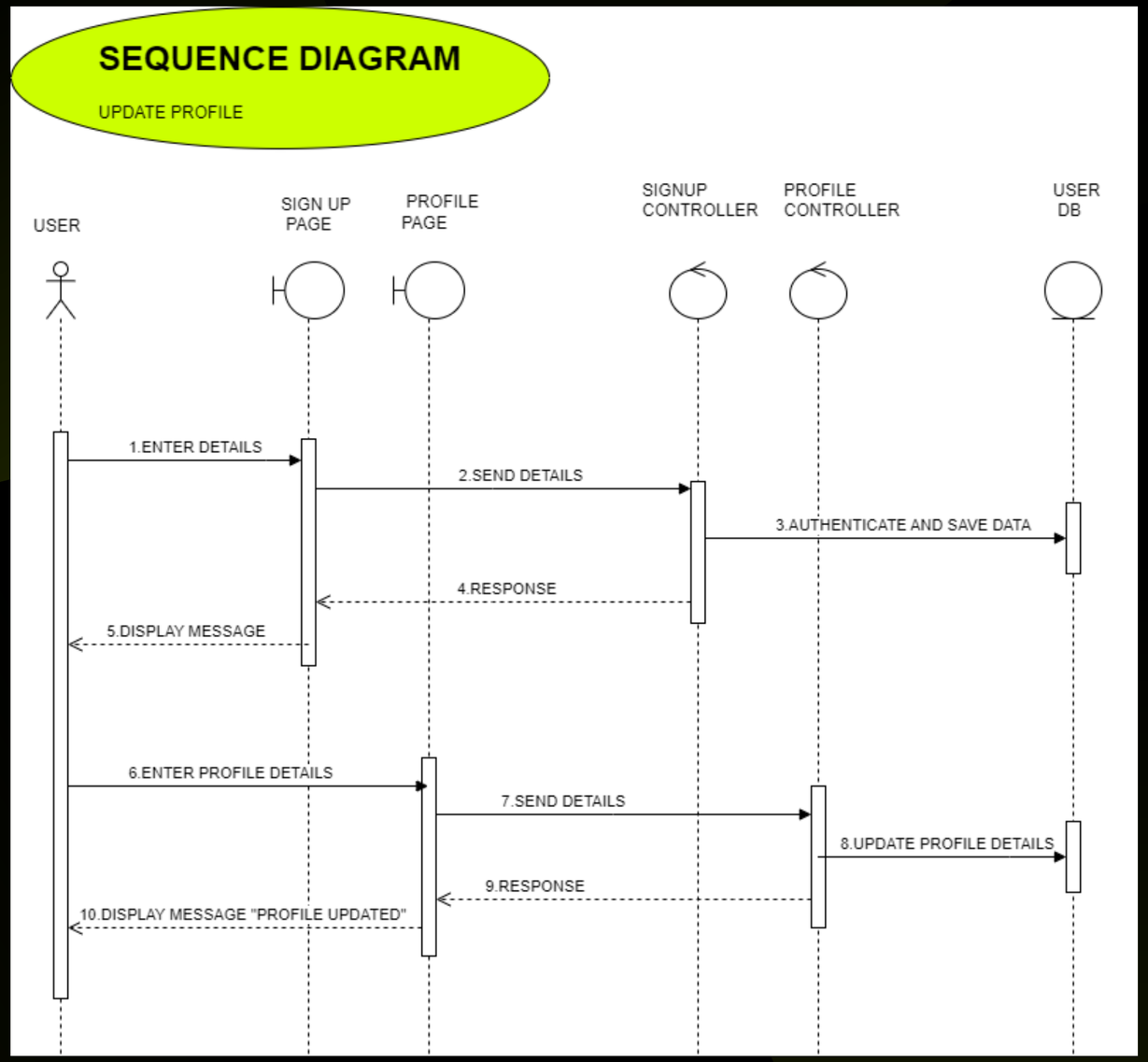

USER
SOCIETY ADMIN
SOCIETY MEMBER
SPONSOR
SIGN UP
CREATE PROFILE
SIGN IN
SEARCH EVENT
REGISTER FOR EVENT
UPLOAD EVENT
MANAGE MEMBERS
CREATE SOCIETY
GET MERCHANDISE
SPONSOR EVENT
LIST SPONSORSHIP PROPOSAL
USER DB
EVENT DB
SOCIETY DB
SPONSOR PROPOSALS DB
USER DETAILS
USER DATA
COMPLETE USER DATA
VERIFICATION STATUS
CREDENTIALS
KEYWORD
EVENT DETAILS
REGISTRATION DETAILS
ACKNOWLEDGEMENT
MEMBER DETAILS
SOCIETY DETAILS
MERCH DETAILS
ORDER DETAILS
MoU
INVOICE
PROPOSAL DETAILS
PROPSALS

DFD

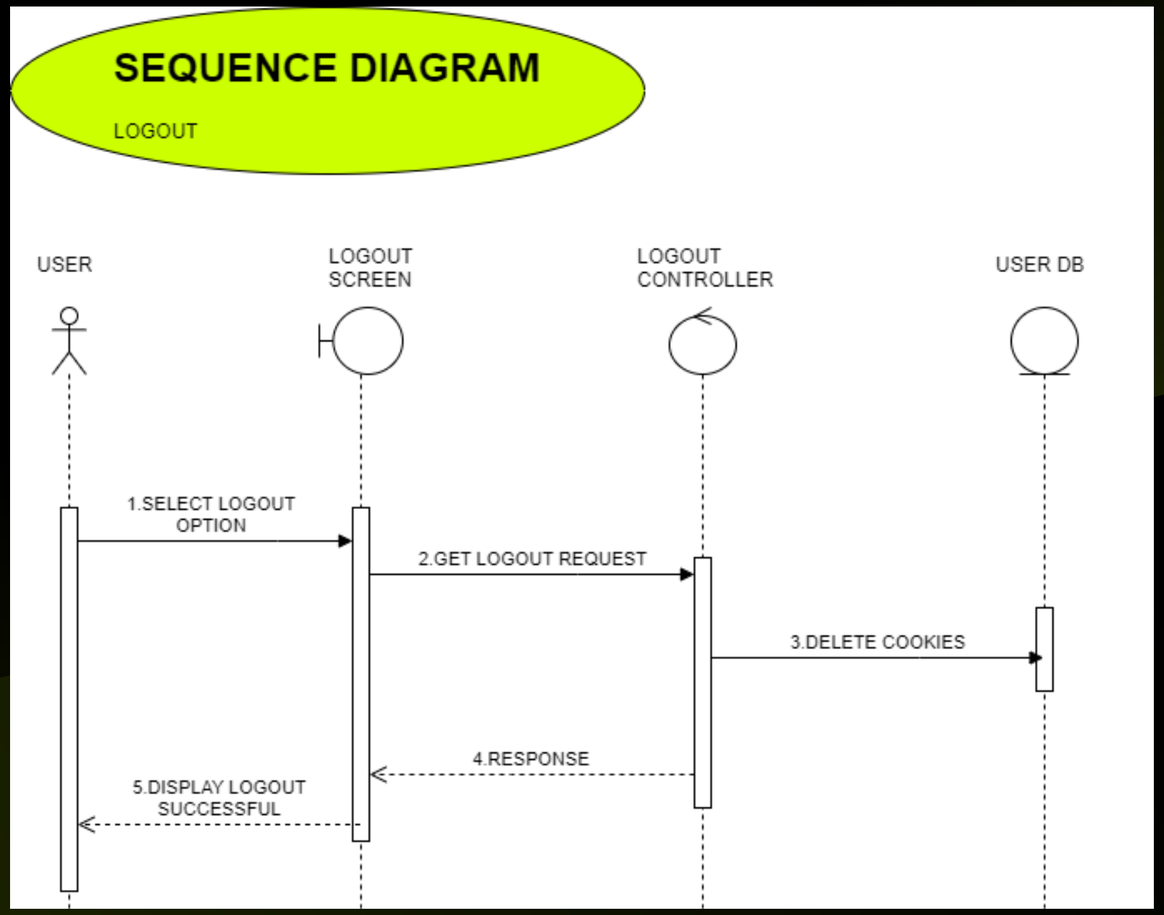
LEVEL - 2


SOCIETY MEMBER
DELETE MEMBER RECORD
UPDATE MEMBER DETAILS
INSERT MEMBER RECORD
UPLOAD EVENT
VALIDATE EVENT
CREATE TASK
ASSIGN TASK
SOCIETY DB
EVENT DB
MEMBER DETAILS
MEMBER DETAILS
MEMBER DETAILS
MEMBER DETAILS
MEMBER DETAILS
MEMBER DETAILS
EVENT DETAILS
EVENT DETAILS
ACKNOWLEDGEMENT
VALIDATED DTAILS
TASK DETAILS
TASK DETAILS
ACKNOWLEDGEMENT
TASK DETAILS

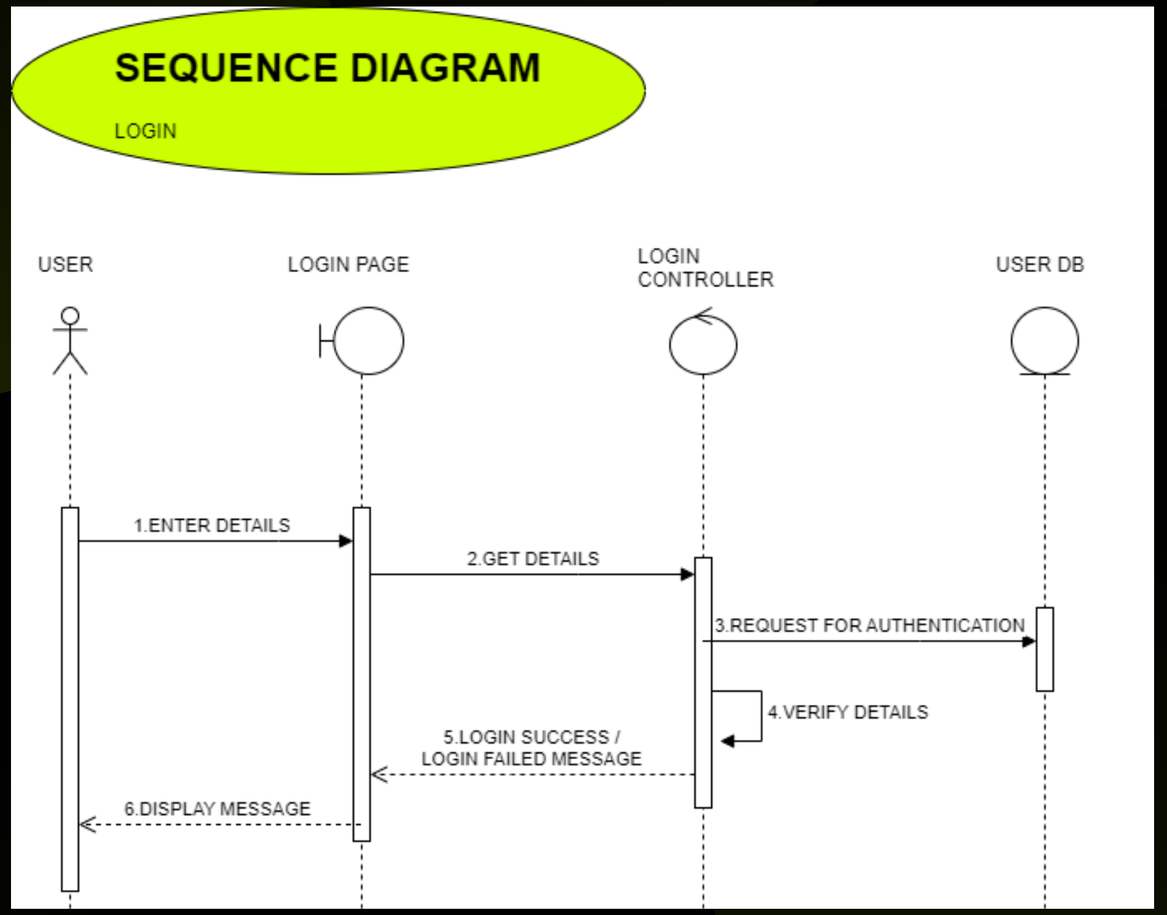
SEQUENCE DIAGRAMS



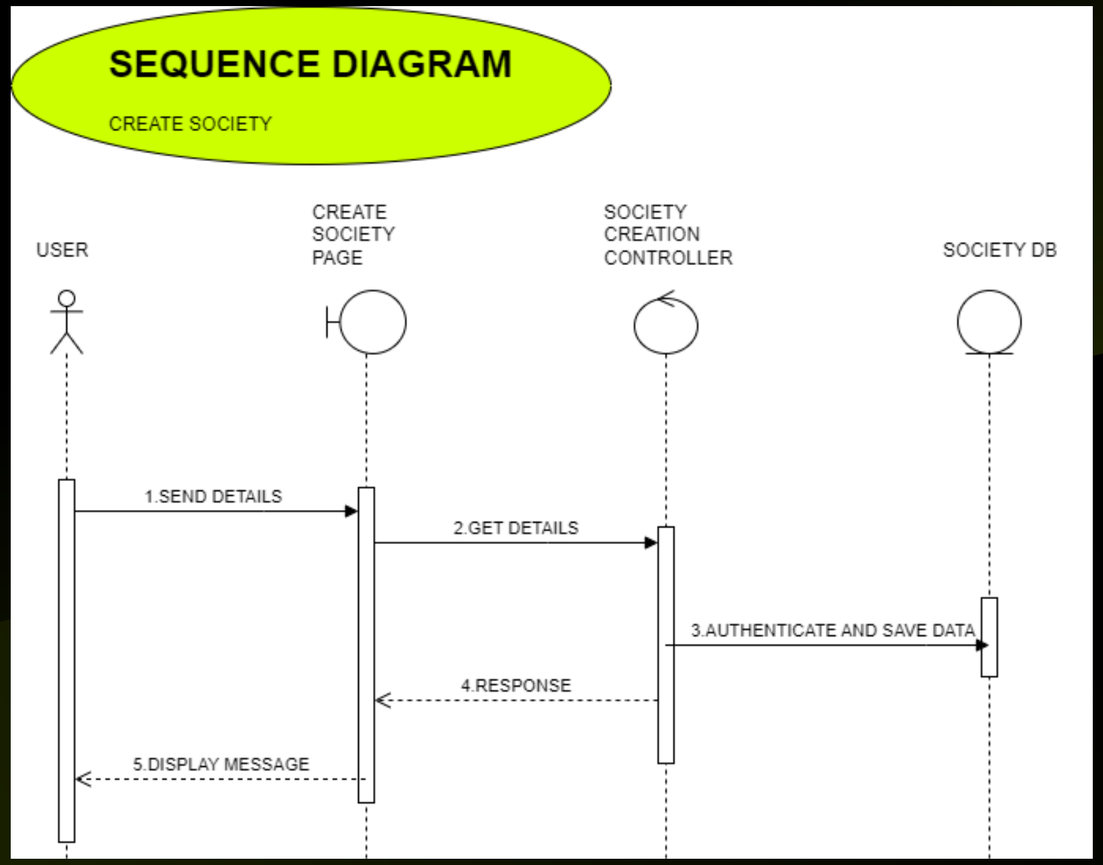
1.


SEQUENCE DIAGRAM
LOGOUT
USER
LOGOUT SCREEN
LOGOUT CONTROLLER
USER DB
1.SELECT LOGOUT OPTION
2.GET LOGOUT REQUEST
3.DELETE COOKIES
4.RESPONSE
5.DISPLAY LOGOUT SUCCESSFUL

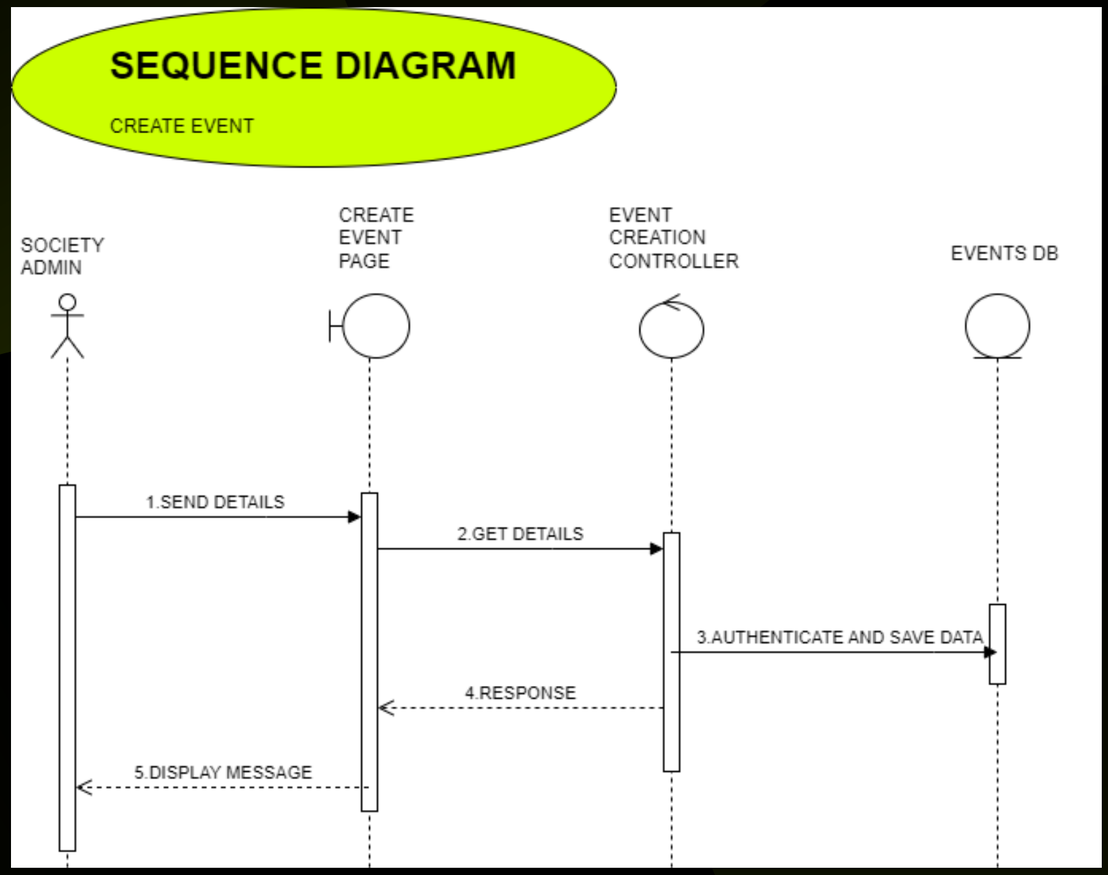
2.


SEQUENCE DIAGRAM
LOGIN
USER
LOGIN PAGE
LOGIN CONTROLLER
USER DB
1.ENTER DETAILS
2.GET DETAILS
3.REQUEST FOR AUTHENTICATION
4.VERIFY DETAILS
5.LOGIN SUCCESS / LOGIN FAILED MESSAGE
6.DISPLAY MESSAGE

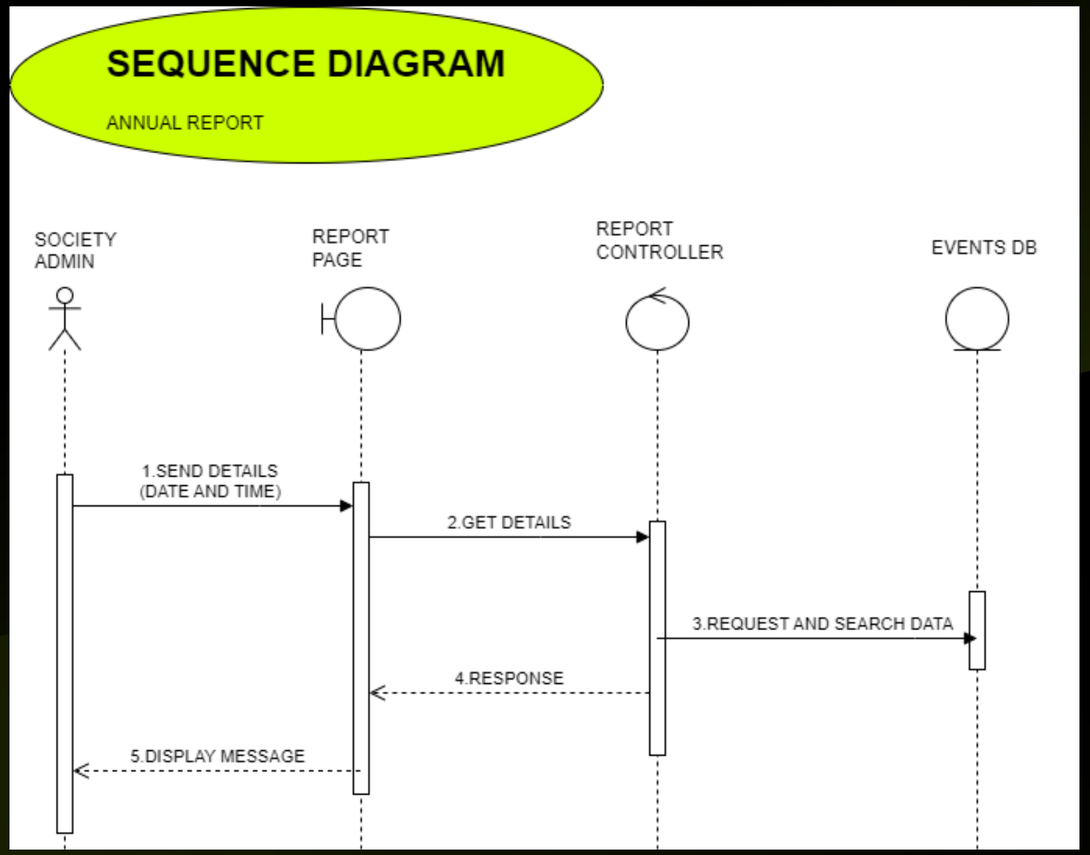
3.


SEQUENCE DIAGRAM
CREATE SOCIETY
USER
CREATE SOCIETY PAGE
SOCIETY CREATION CONTROLLER
SOCIETY DB
1.SEND DETAILS
2.GET DETAILS
3.AUTHENTICATE AND SAVE DATA
4.RESPONSE
5.DISPLAY MESSAGE

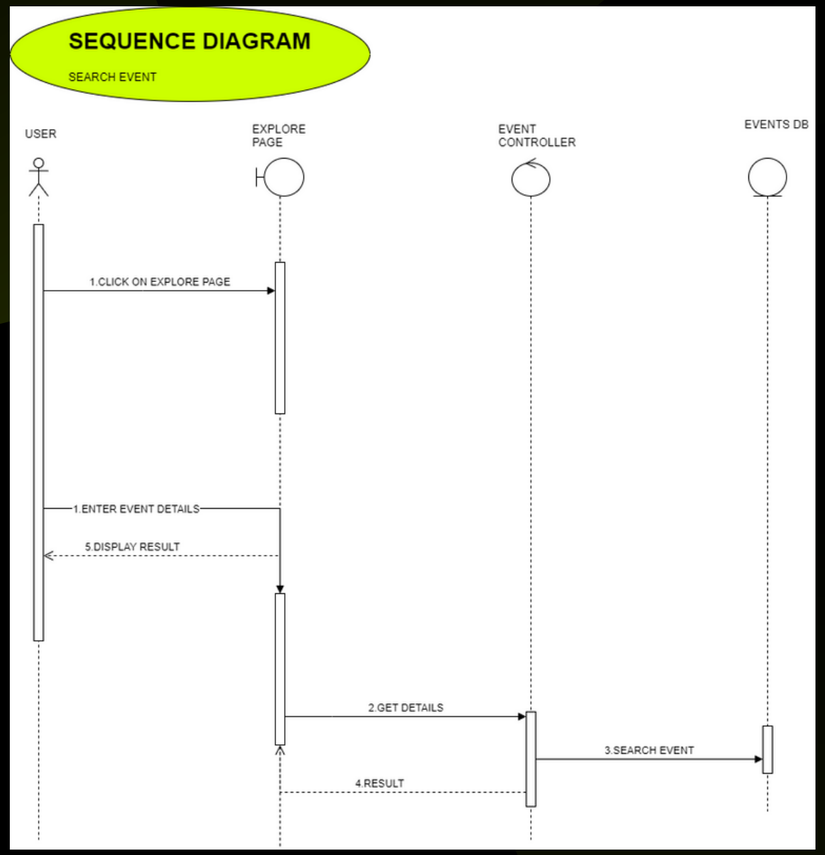
4.


SEQUENCE DIAGRAM
CREATE EVENT
SOCIETY ADMIN
CREATE EVENT PAGE
EVENT CREATION CONTROLLER
EVENTS DB
1.SEND DETAILS
2.GET DETAILS
3.AUTHENTICATE AND SAVE DATA
4.RESPONSE
5.DISPLAY MESSAGE

5.


SEQUENCE DIAGRAM
ANNUAL REPORT
SOCIETY ADMIN
REPORT PAGE
REPORT CONTROLLER
EVENTS DB
1.SEND DETAILS (DATE AND TIME)
2.GET DETAILS
3.REQUEST AND SEARCH DATA
4.RESPONSE
5.DISPLAY MESSAGE

6.


SEQUENCE DIAGRAM
SEARCH EVENT
USER
EXPLORE PAGE
EVENT CONTROLLER
EVENTS DB
1.CLICK ON EXPLORE PAGE
1.ENTER EVENT DETAILS
5.DISPLAY RESULT
2.GET DETAILS
3.SEARCH EVENT
4.RESULT

7.

PSEUDOCODE

```
FUNCTION register_event(event_id, uid):
    IF event_exists(event_id) THEN
        IF NOT is_user_registered(event_id, uid) THEN
            user_details ← get_user_details(uid)
            add_registration(event_id, user_details)
            PRINT "You have successfully registered for the event!"
        ELSE
            PRINT "You are already registered for this event."
        END IF
    ELSE
        PRINT "The event does not exist."
    END IF
END FUNCTION

FUNCTION event_exists(event_id):
    RETURN event_id IN events_db
END FUNCTION
FUNCTION is_user_registered(event_id, uid):
 FOR each registration in registrations_database[event_id] DO
 IF registration['uid'] == uid THEN
 RETURN True
 END IF
 END FOR
 RETURN False
END FUNCTION
```

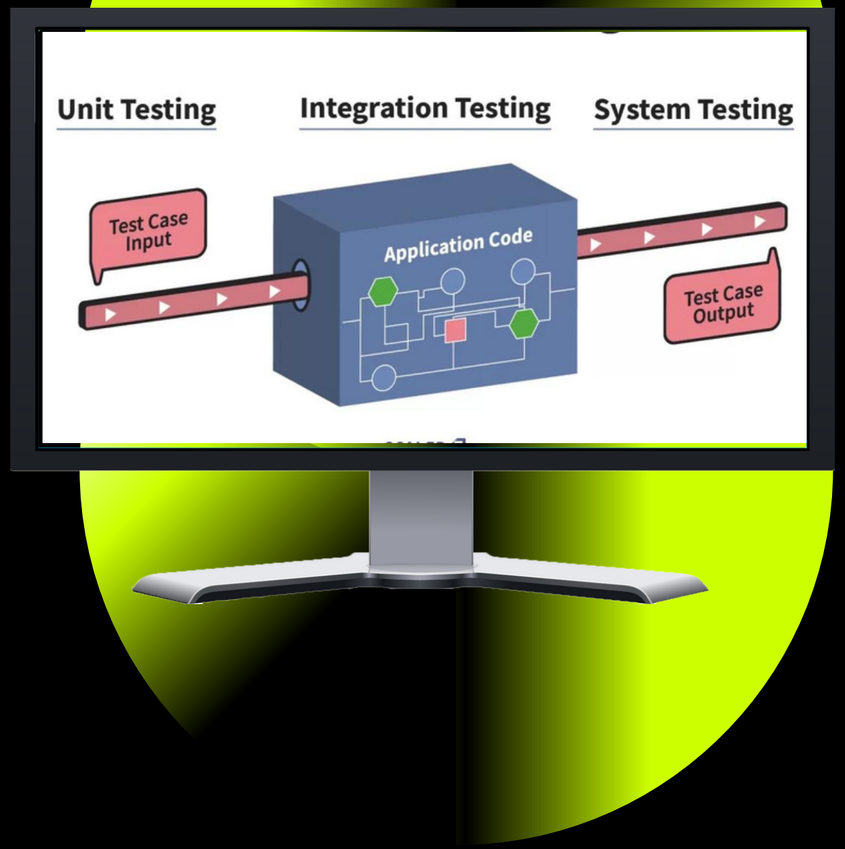
```
FUNCTION get_user_details(uid):
    user_details ← {'uid':uid}
    WHILE True DO
        em = INPUT "Enter your email: "
        IF valid_email(em) THEN
            user_details['email'] ← em
            EXIT WHILE
        ELSE
            PRINT "Please Enter Valid Email"
        END IF
    END WHILE
    user_details['name'] ← INPUT "Enter your name: "

    WHILE True DO
        TRY
            age = INT(INPUT "Enter your age: ")
            IF age == 0 THEN
                RAISE
            END IF
            user_details['age'] ← age
            EXIT WHILE
        EXCEPT
            PRINT 'Please enter valid age'
            PRINT "Please Enter Valid Age"
        END TRY
    END WHILE
```

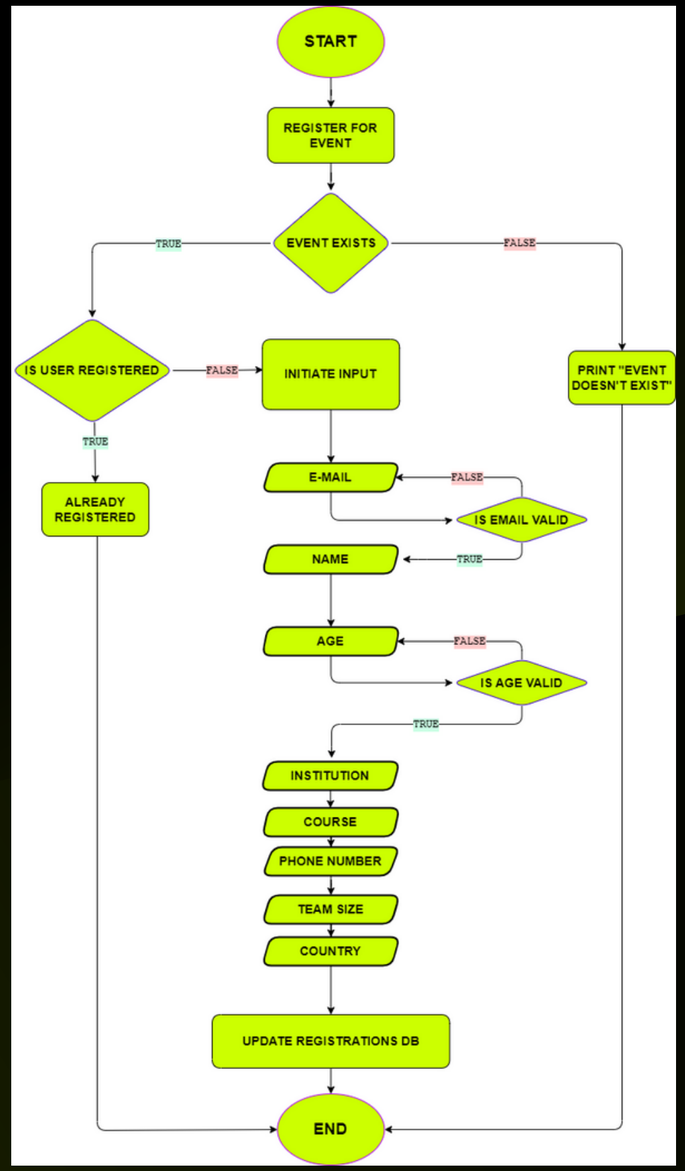
```
    user_details['institution'] ← INPUT "Enter your institution name (if any): "
    user_details['phone'] ← INPUT "Enter your phone number: "
    user_details['course'] ← INPUT "Enter your course (if any): "
    user_details['team_size'] ← INPUT "Enter your team size (if any): "
    user_details['country'] ← INPUT "Enter your country of origin: "
    RETURN user_details
END FUNCTION

FUNCTION add_registration(event_id, user_details):
    APPEND user_details TO registrations_database[event_id]
END FUNCTION

FUNCTION valid_email(email):
    RETURN ("@" IN email) AND ("." IN email)
END FUNCTION
```

WHITE BOX TESTING

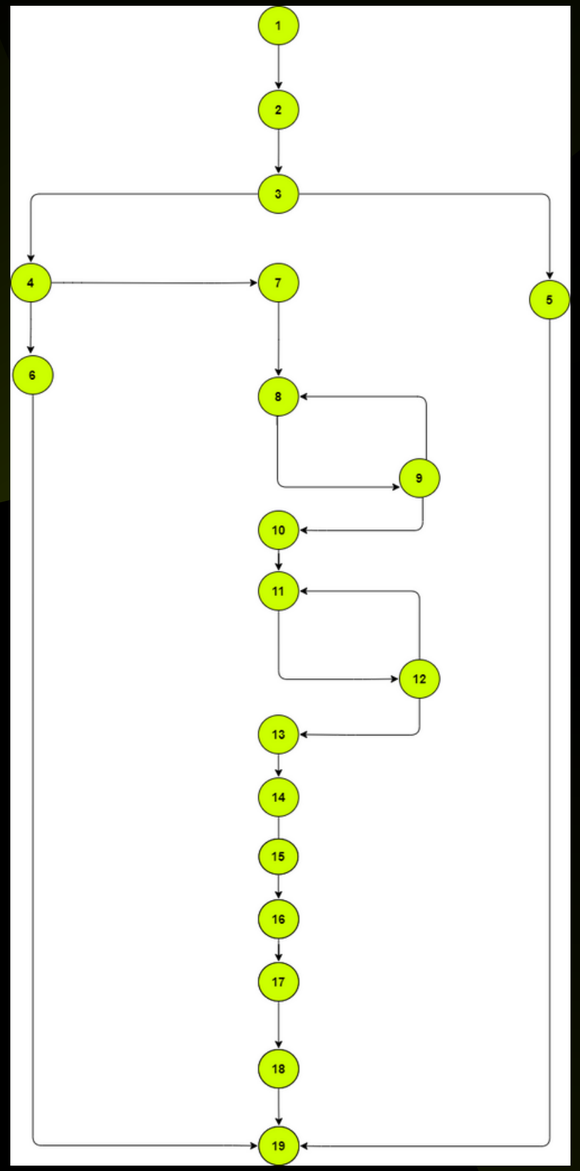
White box testing is a software testing technique that involves examining the internal workings of an application or system being tested. It is also known as clear box testing or structural testing.

During white box testing, the tester has access to the source code of the software being tested and uses this information to design and execute test cases. This approach allows for more in-depth testing of the software's functionality and ensures that all possible paths through the code are tested.





FLOWCHART



FLOWGRAPH

CYCLOMATIC COMPLEXITY

INDEPENDENT PATHS:

i. 1→2→3→4→6→19

ii. 1→2→3→5→19

iii.1→2→3→4→7→8→9→10→11→12→13→14→15→16→17→18→19

iv.1→2→3→4→7→8→9→8→9→10→11→12→13→14→15→16→17→18→19

v. 1→2→3→4→7→8→9→10→11→12→11→12→13→14→15→16→17→18→19

The cyclomatic complexity is a measure of the **number of linearly independent paths** through a program's source code. It can be calculated using the following formula:

M = E - N + 2P

Where:

- M = cyclomatic complexity
- E = number of edges in the flow graph
- N = number of nodes in the flow graph
- P = number of connected components

From the flowchart and the flowgraph, we can count the number of nodes (N), edges (E), and connected components (P) to calculate the cyclomatic complexity of the given pseudocode.

From the flowchart, we can count:

- N = 15
- E = 18
- P = 1

Therefore, the cyclomatic complexity is:

M = E - N + 2P

M = 18 - 15 + 2(1)

M = 5

So the cyclomatic complexity of the given pseudocode is **5.**

THANK YOU

TEAM SOCHUB

Om Gupta - 21078570037

Ashish Durgapal - 21078570007

Om Vaish - 21078570038

Varun Sangai - 21078570055