Department of Computer engineering and software systems
Faculty of Engineering, Ain shams University

PROJECT | CSE334 Software Engineering

SocietySpot

Club Management System

Prepared by

Ahmed Sherif	2201075
Ahmed Wael	22P0221
Ezzeldin Ismail	22P0141
Habiba Sherif	22P0162
Omar Ashry	22P0197
Rawan Hany	22P0180
Rowaida Emad	22P0021
Somaya Ahmed	22P0139

Date

May 8, 2024

Prepared for

Dr. Gamal Ahmed
Eng. Ahmed Zaalouk

TABLE OF CONTENTS

1. software requirement specifications 4
1.1. Introduction 4
1.2. Feasibility Analysis 5
1.3. System Requirements 6
1.3.1. Functional Requirements 6
1.3.2. Non-functional requirements 6
1.4. Requirements Validation 7
1.4.1. Requirement traceability matrix 7
1.4.2. Source traceability matrix. 9
1.5. Time Plan 10
1.6. Process Model 11
1.7. Conclusion 11
2. ANALYSIS AND DESIGN DOCUMENTATION 12
2.1. Use Case Diagrams 12
2.1.1. Use Case Description 14
2.2. Swim Lane Diagrams 19
2.3. Sequence Diagrams 24
2.4. State Diagrams 28
2.5. Data Flow Diagrams (DFDs) 29
2.6. System Architecture 37
2.6.1. Architecture Style 39
2.7. Component Diagram 41
2.8. Class Diagram 43
3. TESTING AND USER GUIDE 46
3.1. Testing Strategy 46
3.2. End-User Guide 48
3.3. Conclusion 51
REFERENCES 52

LIST OF FIGURES

Figure 1 Time Plan.....10
Figure 2 Student Use Case Diagram.....12
Figure 3 User Management Use Case diagram.....13
Figure 4 Club Leader, Member Use Case diagram.....13
Figure 5 Add Post swim Lane diagram.....19
Figure 6 Join Club swim Lane diagram.....20
Figure 7 Create a Club Swim Lane diagram.....21
Figure 8 Login swim Lane diagram.....22
Figure 9 Create an account Swim Lane diagram.....23
Figure 10 Club member Sequence diagram.....24
Figure 11 Club leader Sequence diagram.....25
Figure 12 University administrator Sequence diagram.....26
Figure 13 Student Sequence diagram.....27
Figure 14 Student State diagram.....28
Figure 15 University administrator State diagram.....28
Figure 16 Context Level DFD diagram.....29
Figure 17 Level 0 DFD diagram.....30
Figure 18 Level 1 DFD For Process 1.....31
Figure 19 Level 1 DFD For Process 2.....32
Figure 20 Level 1 DFD For Process 3.....33
Figure 21 Level 1 DFD For Process 5.....34
Figure 22 Level 1 DFD For Process 4.....34
Figure 23 Level 1 DFD For Process 7.....35
Figure 24 Level 1 DFD For Process 6.....35
Figure 25 Level 1 DFD For Process 9.....36
Figure 26 DFD To Architecture diagram.....38
Figure 27 Data-centered architecture diagram.....39
Figure 28 Merged architecture.....40
Figure 29 Component diagram.....41
Figure 30 Class diagram.....45

1. software requirement specifications

1.1. Introduction

SocietySpot is leading the way in transforming the university club's scene; our platform is suitable for a wide range of student populations such as club presidents, participants, and University sports managers. SocietySpot's goal is to create a lively campus culture that will be accomplished by offering an inclusive space where individuals from different walks of life meet and work together in harmony.

For students, SocietySpot serves as a gateway to exploration and engagement. With a Club Directory feature, students can effortlessly discover and join clubs once they identify the major they want to join. Students can explore detailed profiles of each club, including their mission statements, and activities. Registering for events and signing up for event-related activities. SocietySpot provides each student with a dedicated profile, allowing them to showcase their interests, club affiliations, and contributions. Through personalized engagement and networking opportunities, SocietySpot enhances the university community experience.

As club members, they engage with their clubs and participate in club activities. The system empowers club members by interacting with club leaders through discussions and voting polls features. Additionally, SocietySpot integrates gamification elements such as points and badges that club members can earn for their participation and interaction.

for club leaders, SocietySpot will enable club leaders to establish and manage their clubs within the university community. Club leaders can approve or deny membership requests, track member attendance and participation, and communicate with club members regarding club activities and updates. Club leaders can create event listings, manage event details and logistics, and promote events to club members and the wider university community.

Finally, University administrators play a pivotal role in ensuring the success and integrity of the club ecosystem. Administrators have access to administrative dashboards and tools that provide insights into club creation, membership management, and event planning.

SocietySpot enables administrators to derive valuable insights from club engagement data. By analyzing metrics such as membership trends, and event attendance, administrators can make informed decisions, and identify areas for improvement.

SocietySpot is an innovative tool created to convert how clubbing is done in the university. This website was created for students who want up-to-date information about the clubs they belong to, their leaders, or their administrators. It was made with students in mind mainly by its members. In addition, it is a comprehensive package meant to simplify the management of various clubs, streamline communication between members, and enhance participation levels.

1.2. Feasibility Analysis

Feasibility analysis for the SocietySpot system involves assessing its technical, economic, operational, and scheduling feasibility to determine whether the system is viable and beneficial to develop and implement. From the technical angle, evaluating whether the technical requirements for the system including server capacity, network bandwidth, and security protocols can be supported by the current infrastructure of the university. Additionally, check to see if it works with another type of university website, including student information systems and learning management systems, so that information can easily be transferred between them.

From the economic and financial angle, estimate the cost of formally developing, installing, and maintaining SocietySpot to include a software program and hardware for procurement/use licensing which attracts ongoing support charges. This includes mapping out any potential risks associated with the project's financial viability, that may emanate from cost overruns or shortfalls in revenue or any unanticipated cost and developing relevant strategies for mitigating them.

For the operational feasibility, to gauge their readiness to implement and use SocietySpot, the stakeholders, who include students, club managers, and university administrators should be polled or made to participate in focus groups. Also, assess users' training structures to guarantee that they possess all that is supposed to work well with the software (User Acceptance Testing for SocietySpot).

1.3. System Requirements

1.3.1. Functional requirements

1.1. Students and Administrators can create personal accounts.

1.2. Club members can join new clubs.

1.3. Students and administrators can sign up or log in to their accounts.

1.4. Club leaders can establish clubs.

1.5. Club leaders can post updates.

1.6. Club leaders can approve club members.

1.7. Club members can use online forms to discuss upcoming events, share resources, and collaborate on projects.

1.8. Club leaders can give rewards to encourage participation in club events.

1.9. Administrators are allowed to visit the dashboard of club members and leaders to view their activity levels.

1.10. Club leaders create voting polls.

1.11. Club leaders can manage and customize pages for clubs.

1.3.2. Non-functional requirements

2.1. Accounts are secured with a special username and password.

2.2. The system should have an intuitive and user-friendly interface for ease of use.

2.3. The system should be available and accessible to the user all the time.

2.4. Data stored should be consistently accurate and up to date.

2.5. The system should regularly backup user data to prevent loss in case of system failure or data corruption.

2.6. The website should support multiple languages in case of foreign users.

2.7. The system should include feedback mechanisms and feedback collected should be used to identify areas for improvement.

1.4. Requirements Validation

This phase checks the requirements for realism, consistency, and completeness. errors in the requirements are identified and corrected in this phase requirements are technically reviewed by the review team which includes software engineers, customers, and end-users.

1.4.1. Requirement traceability matrix

This requirements traceability matrix describes the dependency relationship between the requirements.

SocietySpot requirements traceability matrix:

Req. ID.	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	1.10	1.11
1.1			✓								
1.2	✓		✓	✓		✓					
1.3											
1.4	✓		✓								
1.5	✓		✓	✓							
1.6	✓	✓	✓	✓							
1.7	✓	✓	✓	✓		✓					
1.8	✓		✓								
1.9	✓		✓								
1.10	✓		✓								
1.11	✓		✓	✓							

Table 1, Requirements Traceability Matrix

Assumptions:

1. Users creating new personal pages: Personal pages can be created after signing up (by entering their details).

2. Students can join new clubs: To join new clubs, Students need to have a personal page and log in to it or to sign up. Additionally, clubs must be established by the club leader and the club member must be approved.

3. Students and administrators can sign up or log in to their accounts: This requirement is independent.

4. Club leaders can establish new clubs: Club leaders need to have a personal page and log in to it or sign up.

5. Club leaders can post updates: Club leaders need to have a personal page and log in to it or sign up. They also need to create a club.

6. Club leaders can approve new members: Club leaders need to have a personal page and log in to it or sign up. Students must join the club so that they can be approved.

7. Club members can use online forms to discuss upcoming events, share resources, and collaborate on projects: Club members need to have a personal page and log in to it or to sign up. They also need to join clubs and get approved by the club leader.

8. Club leaders can give rewards to encourage participation in club events: Club leaders need to have a personal page and log in to it or sign up.

9. Administrators are allowed to visit the dashboard of club members and leaders to view their activity level: Administrators need to have a personal page and log in to it or sign up.

10. Club leaders create voting polls: Club leaders need to have a personal page and log in to it or sign up.

11. Club leaders can manage customized pages for clubs: Club leaders need to have a personal page and log in to it or sign up. They need to create the club before they customize or start to manage it.

All non-functional requirements are independent.

1.4.2. Source traceability matrix.

The source traceability matrix shows the links from requirements to stakeholders who proposed these requirements.

SocietySpot source traceability matrix:

Req. ID	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	1.10	1.11	2.1	2.2	2.3	2.4	2.5	2.6	2.7
Administrator	✓		✓						✓			✓	✓	✓	✓	✓	✓	✓
Club leader	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	
Club member	✓	✓	✓				✓					✓	✓	✓	✓	✓	✓	

Table 2, Source Traceability Matrix

Assumptions:

1. Administrator: Administrators can view any club member's / leader's activity.

2. Club leaders/members: Both club leaders and members are students, and each member can start a new club then he will be the leader of the club he established. Besides, they can join any other established club as normal club members.

1.5. Time Plan

In this section, we will propose a SocietySpot web application project plan Encompassing the primary tasks to be completed. The plan includes code implementation and documentation. The subsequent table illustrates the tasks of the project and the individuals responsible for each task. Furthermore, the chart demonstrates the project plan.

Id	Task	Start Date	Number of Days	Team Member
1	Create frontend	1-May	7 days	Omar, Ezz, Habiba
2	Create backend	26-Apr	11 days	Ezz
3	Documentation report	6-May	2 days	Rawan, Ezz
4	Matrices	14-Apr	3 days	Somaya, Ahmed Wael
5	UML diagrams	17-Apr	9 days	All team members

Table 3, Time Plan

Figure 1 Time Plan

1.6. Process Model

Our System used the Waterfall process model as its requirements are definitive and it is suitable for big projects which have several team members.

- Waterfall Process Model: it is the breakdown of the project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialization of tasks.

- It is flexible as progress flows in largely one direction
- It has the following phases:

1.7. Conclusion

In conclusion, the SocietySpot web application represents an innovative solution that has the potential to transform the university club scene, fostering a vibrant and inclusive campus culture. With continued collaboration, feedback, and iteration, we believe that SocietySpot will become a valuable asset to the university community, facilitating engagement, collaboration, and growth among students and club members alike.

2. ANALYSIS AND DESIGN DOCUMENTATION

2.1. Use Case Diagrams

Figure 2 Student Use Case diagram.

Figure 3 User Management Use Case diagram.

Figure 4 Club Leader, Member Use Case diagram.

2.1.1. Use Case Description

Use case description for creating a new account:

Use case name: Create an account.

Related Requirements: Requirement 1.3.
Students and University supervisors can sign up or log in to their accounts.

Goal in context: A new user requests a new personal page from the university supervisor.

Preconditions: Users requesting a new account have appropriate proof of identity.

Successful end condition: A new account is created for the user.

Failed end condition: Application for new account is rejected.

Primary actors: User.

Secondary actors: University supervisor.

Trigger: The user asks the university supervisor to create a new account.

Main flow:

Step	Action
1	The user asks the university supervisor to create a new account.
2	Users select an account type.
3	The user enters the author's details.
4	The user details are verified using the system database.
5	The new account is created.
6	A summary of the new account is emailed to the user.

Extensions:

Step	Branching action
4.1	The database does not accept the user's details.
4.2	The user's account application is rejected.
4.3	Authorization.

Use case description for login:

Use case name:	Login.
Related Requirements:	Requirement 1.1. The user must have an account.
Goal in context:	The user logs into his account.
Preconditions:	The user has to access the login page in the system.
Successful end condition:	Account accessed.
Failed end condition:	Account denied.
Primary actors:	User.
Secondary actors:	Database.
Trigger:	The login form appears to the user.

Main flow:

Step	Action
1	User starts up the system.
2	Users select an account type.
3	The user enters their personal information (username and password).
4	The user's info is verified by the system database.
5	Account is accessed by the user.

Extensions:

Step	Branching action
4.1	Incorrect information entered.
5.1	Access denial.

.

Use case description for creating a club:

Use case name: Create a club.

Related Requirements: Requirement 1.1, Requirement 1.3.
Users must have an account and can log in to it or sign up.

Goal in context: The club leader creates a new club for club members to join.

Preconditions: The club leader has been promoted to supervise the club.

A successful end condition: Club is created.

Failed end condition: Club creation is denied.

Primary actors: Club leader.

Secondary actors: Club member.

Trigger: Receiving a significant number of requests from the club
members interested in forming a club.

Main flow:

Step	Action
1	Club leader requests to create a new club from the university supervisor.
2	Club leader customizes new club.
3	The club leader gives a name and a logo for the club.
4	Club members request to join the club.
5	Club members are verified by the club leader.
6	Club is Created.

Extensions:

Step	Branching action
1.1	The university supervisor rejects creating a new club.
4.1	Club members are rejected.
6.1	**Club leaders can promote members.**
6.2	**Club leaders make events.**
6.3	**Club leader adds and removes members.**

Use case description for joining a club:

Use case name:	Join club.
Related Requirements:	Requirement 1.1, Requirement 1.3, Requirement 1.4, Requirement 1.6. Club members need to have a personal page and log in to it or sign up. Also, clubs must be established by the club leader and the club members must be approved.
Goal in context:	Club members join clubs established by club leaders.
Preconditions:	Club members have accessed the homepage containing available clubs.
Successful end condition:	Club member approved to join.
A failed end condition:	Club member is rejected.
Primary actors:	Club members.
Secondary actors:	None.
Trigger:	Club member requests to join a club.

Main flow:

Step	Action
1	Club member selects a club option from the system.
2	Club member requests to join the club.
3	Club member is verified by the club leader.
4	Club member joins the club.

Extensions:

Step	Branching action
3.1	Club member is rejected.
3.2	The club leader does not accept the club member's details.

Use case description for adding a new post:

Use case name:	Add post.
Related Requirements:	Requirement 1.1, Requirement 1.3, Requirement 1.4. The club leader needs to have a personal page and log in to it or sign up. Also, clubs must be established by the club leader to be allowed to post.
Goal in context:	User posts updates and new announcements on the club.
Preconditions:	The user should have access to the club.
Successful end condition:	A new post is added.
Failed end condition:	Post is rejected and can't be posted.
Primary actors:	User.
Secondary actors:	None.
Trigger:	The user composes their new post and then submits it for review or publication.

Main flow:

Step	Action
1	User provides their login credentials or uses alternative authentication methods.
2	Users access the club platform.
3	Users create and submit their new posts.
4	Posts are reviewed and verified by the administrator.
5	Posts have been added.

Extensions:

Step	Branching action
1.1	Authentication failed.
2.1	Users can't access the club.
4.1	The university supervisor rejected the post.
4.2	Post is denied.

2.2. Swim Lane Diagrams

Figure 5 Add Post swim Lane diagram.

Figure 6 Join Club swim Lane diagram.

Figure 7 Create a Club swim Lane diagram.

Figure 8 Login swim Lane diagram.

Figure 9 Create an account Swim Lane diagram.

2.3. Sequence Diagrams

Figure 10 Club member Sequence diagram.

Figure 11 Club leader Sequence diagram.

Figure 12 University administrator Sequence diagram.

Figure 13 Student Sequence diagram.

2.4. State Diagrams

Figure 14 Student State diagram.

Figure 15 University administrator State diagram.

2.5. Data Flow Diagrams (DFDs)

- **Context Diagram**

Figure 16 Context Level DFD diagram.

- **Level 0 Diagram**

Figure 17 Level 0 DFD diagram.

- **Level 1 Diagrams**

Figure 18 Level 1 DFD For Process 1

Figure 19 Level 1 DFD For Process 2

Figure 20 Level 1 DFD For Process 3

Figure 21 Level 1 DFD For Process 4

Figure 22 Level 1 DFD For Process 5

Figure 23 Level 1 DFD For Process 6

Figure 24 Level 1 DFD For Process 7

Figure 25 Level 1 DFD For Process 8

Figure 26 Level 1 DFD For Process 9

2.6. System Architecture

Figure 27 DFD To Architecture diagram.

2.6.1. Architecture Style

- **Data Centered Architecture**

Figure 28 Data centered architecture diagram.

Figure 29 Merged architecture.

2.7. Component Diagram

Figure 30 Component diagram.

Explanation:

1- The user enters their login information to access their account.

2- The user sets up their profile page with their information which is stored in a database.

3- The user can choose whether to create a club or search for a club

4- If the user chooses to create a club, approval from the respective university is required.

4.1- User enters club details to create a club dashboard which is stored in a database

4.2- User becomes club leader once the club is created

4.3- club leaders can view the performance dashboard of the club

4.4- club leaders can make announcements on the announcement dashboard (voting polls, discussions, events)

4.5- club leaders can access the chatroom (sending and receiving texts to and from club members)

4.6- club leaders can view requests made from users wanting to become club members

4.7- club leaders can approve or reject requests.

4.8- club leaders can access club members' data to update their role

5- If a user searches for a club, the search is sent to the club's database which returns search results

5.1- users can request to join the club.

5.1.1- After the user is approved by the club leader, their data is stored in the club members database and the user becomes a club member.

5.2- club members can view the club activity dashboard

5.3- club members can view the Announcement dashboard

5.4- club members can access the chatroom (sending and receiving texts to and from other club members and club leader)

2.8. Class Diagram

<<interface>>
User
- username : String
- name : String
- email : String
- password : String
- age : int
- enrollmentRequestStatus : boolean
- userType : String
+ User(username: String, enrollmentRequestStatus: boolean, age: int, email: String, name: String) : void
+ getName() : String
+ getUsername() : String
+ getEmail() : String
+ getPassword() : String
+ getAge() : int
+ setStatus(status: boolean) : void
+ makeEnrollmentRequest(requester: User) : Request
+ updateEmail(newEmail: String) : boolean
+ updatePassword(newPassword: String) : boolean
Request
-user:User
-approve:boolean
+MakeRequest(usRequester:User):
+ApproveRequest():void
+getRequestStatus():void
make
1
1
contain
0..*
1
UserManeger
-UsersInMyApp:List<User>
-allEnrollementRequests:List<Request>
+addUser(userInfo : User) : User
+deleteUser(userId : String) : void
+updateUser(userId : String):void
+FindUser(username:String):User
+Login(Username,Password):void
+signup():void
+addRequest(user,User):void
+deleteRequest(request:Request):void
+getAllRequests():List<Request>
+getNumberOfRequests():int
Vote
- voteId : String
- user : User
- voteType : boolean
- voteDate : Date
+ Vote(voteId: String, user: User, voteType: boolean, voteDate: Date) : void
+ getVoteId() : String
+ getUser() : User
+ getVoteType() : boolean
+ setVoteType(voteType: boolean) : void
+ getVoteDate() : Date
+ changeVoteType(newVoteType: boolean) : void
0..*
make
0..*
contain
1
Event
eventName : String
participants : List<User>
hostClub : Club
Event(Name:String,hostclub:Club)
getEventDetails():void
showHostClub():void
Post
-postId : String
-content : String
-author: User
-creationDate : Date
-votes: Lisr<Vote>
+Post(user:User)
+editPost(newContent : String):void
+ViewPost():void
+addVote(type:boolean):void
+gellAllvotes():List<vote>
+ deleteVote() : void
+ changeVote(Vote: boolean) : void
0..*
make
0..*
contain
:Student
-bio : String
-University : String
-ProofOfUniEnrollment :bool
- joinedClubs : List<Club>
+SetProfilePage() : void
+ searchClubs(searchTerm: String) : List<Club>
+ joinClub(club: Club) : Void
+ removeClub(club: Club) : boolean
+ getNumberOfClubs() : int
+ getClubs() : List<Club>
+ isMemberOfClub(club: Club) : boolean
+getProofOfUniEnrollment() :bool
:UniverstySupervisor
-deparment:String
+ApproveEnrollementRequest():void
+showAllRequests():List<Request>
+deleteRequest():void

Figure 31 Class diagram.

3. TESTING AND USER GUIDE

In this section, we will view the SocietySpot website and verify the functionality.

3.1. Testing Strategy

Since SocietySpot has no request handlers, our system front-end could be tested using (postman) and for testing the whole implementation we can just check the change of database upon any change from any update.

Front-end testing:

1. creating a club:

Checking the list of posts:

http://localhost/swe/admin/viewposts.php
GET
Send
Query Params
List of Posts
ID
Title
Content
User ID
Created At
1
hi
hihihihihihihihihihihihihih
1
2024-05-07 00:32:38
2
sba7o
sabh sabh
9
2024-05-07 00:33:09
3
abyaaaaaaaaaaaaaad
aaaaaaaaaaaaaa

1. Test Scenario 1: Sign up for new accounts to verify the change in the database.

2. test Scenario 2: trying to sign up for an existing account.

Test case ID	Test case description	Test data	Expected result
1	Saving new accounts to the database.	Information is saved to the database with a hashed password.	Data is stored in the database.
2	Trying to save a new account with the information of the existing one.	Didn't overwrite, nor delete old data or lead to errors.	Printing error duplicated entry.

Test scenario 1:

We can see that the user data is stored successfully.

Test scenario 2:

No unexpected errors occurred.

3.2. End-User Guide

Fill in the required information then select a role (admin, student, club member, club leader)

After signing up you can log in leading to the profile

The profile will view all the functions

for a sample, we can view the admin dashboard:

Creating a post:

3.3. Conclusion

Throughout this report, we have outlined the software requirement specifications, conducted a feasibility analysis, defined system requirements, and validated those requirements through traceability matrices. We have also provided a detailed time plan for project execution and discussed the chosen process model, which is the Waterfall model, for managing the development process.

Furthermore, we have delved into the analysis and design documentation, including use case diagrams, swim lane diagrams, sequence diagrams, state diagrams, data flow diagrams, system architecture, and component diagrams. Each of these components contributes to the comprehensive understanding and visualization of the SocietySpot system. Additionally, we have outlined the testing strategy, and user guide, and provided a demonstration of the end-user experience. Through thorough testing and user guidance, we ensure the functionality and usability of the SocietySpot web application.

In summary, the SocietySpot project holds immense potential to revolutionize how university clubs operate and interact. With continued collaboration, feedback, and refinement, we are confident that SocietySpot will become an invaluable asset to the university community, enhancing engagement, communication, and collaboration among students, club leaders, and administrators.

REFERENCES

[1] Software Engineering: A Practitioner's Approach, 9th Edition

[2] https://guides.visual-paradigm.com/creating-sequence-diagrams-with-a-use-case-driven-approach-a-comprehensive-guide/

[3] https://app.diagrams.net/index.html

[4] https://www.visual-paradigm.com/

[5] https://www.lucidchart.com/pages/data-flow-diagram