Contents

1. Contents + Contributions Table
   Chapter 1
2. Requirements Gathering Methodology
3. Reflections on Methodology and Results
4. User Requirements
5. Concepts – System Requirements
6. Functional Requirements – System Requirements
7. Functional + Non-Functional Requirements – System Requirements
   Chapter 2
8. Architecture Model + Discussion
9. Use Case Diagram + Models + Discussion
   Chapter 3 + Chapter 4
10. Implementation + Testing Discussion
    Chapter 5
11. Critical Analysis
    Appendix – Participant Forms + Research Transcripts, Progress Tracker

Contributions Table

Team member name	Detailed contributions per chapter	Percentage contributions overall	Signature
Michael Parker UP2263259	Chapter 1: Problem specification Chapter 2: Design Chapter 3: Implementation Chapter 4: Testing Chapter 5: Critical analysis	33.33%	
Lewis Ryan UP2277906	Chapter 1: Problem specification Chapter 2: Design Chapter 3: Implementation Chapter 4: Testing Chapter 5: Critical analysis	33.33%	
Nickyyl Sharma UP2269939	Chapter 1: Problem specification Chapter 2: Design Chapter 3: Implementation Chapter 4: Testing Chapter 5: Critical analysis	33.33%	

Our Problem

The problem we are trying to solve is the struggle society members experience in accessing information and interacting with their society. We propose an app that collects social news and places it into a single platform. This app will also sync societies' calendars and events to users' phones. Additionally, the app will allow users to RSVP, raise queries to the committee, and receive alerts about any changes to events they have RSVPed to. There will also be a system to like and comment on posts made by the committee.

Requirements Gathering Methodology

To develop requirements, we utilised a semi structured interview, with an additional set of questions designed for society committee members – collecting qualitative primary data. This allowed us to gain a comprehensive understanding of our participants' experience as both members of a society, and members of their committees.

We made use of an opportunity sampling method – and for the majority of our interviews we attempted to keep our sample representative by limiting our sample frame to people sitting down in the following buildings: Eldon, Park, University Library

We interviewed 11 members (4 of whom were on a committee) who were members of a wide range of societies, on average for 7 minutes each, recording each one as it happened. The interviews were then transcribed for us to analyse, allowing us to see patterns to inform our requirements.

We later carried out some product reviews, informing our design approach – based on the key competitors in the market.

Interview 11 was conducted with an alternative question set as it was completed pre-split. This served as the basis of our below question sets, designed for more useful results.

Question Sets

1. How many societies are you a part of?
2. How many different apps do you have for society news?
- Would you be interested in something to put all of these in one place?
3. When you are missing information, how do you get in touch with your societies?
- How long does it take for them to get back to you?
4. Do you find you miss events due to messages getting lost?
5. What type of posts do the societies make on social media?
6. What is your favourite social media app that you use for societies and why?
7. How many social media apps do you use, and of these which do you use the most and why?
8. How many societies where you interested in, and how many have you actually gone to?
- Why did you miss their events?
9. What's your best way to interact with a society's event?
10. Do you prefer seeing the details first, chatting to someone before deciding or being able to

reserve a spot through an app?
11. If you were to share an event with friends, what would be the easiest way for you to do that?
12. If you're deciding whether to attend an event, does viewing a guest list help?

1. How do distribute news to your members?
- Why do you use this method
2. How would you say members engage with your announcements?
3. What is the biggest hurdle to engage with your society?
4. What forms of media (E.g. text, images, video, polls) do you distribute and why?
5. Have you set up a shared calendar? If not, do you believe this could be useful?
6. Do you send out communication over more than one channel already?
7. Do you manage your society on multiple social media accounts, if so, how difficult is this
8. When managing queries related to the society, how long do you typically take to reply
9. When posting about events, what information do you typically include?
10. What is your typical event turnout, and how do you think it might be improved?

Reflections on Methodology

Largely we believe our methodology to have been successful, through the nature of our semi structured interview format, we avoided many of the pitfalls of structured and unstructured interviews. The use of an opportunity sample may have been problematic, as it would have given rise to investigator effects and so our data may not be as representative as we hope, potentially lacking external validity.

On the other hand, our research may also lack internal validity, with our 3rd question perhaps being too leading – giving way to demand characteristics; with all our participants replying positively.

Whilst most of the individual questions were suitable for prompting participants, question 11 proved to be less useful, as finding out the details was ultimately a must for all participants – and so we would change this if we did this again.

Our 9th society member question was also too vague, requiring us to add clarification which should have been there to begin with.

Reflections on Results

As mentioned above, all participants were positive (though Interview 5 had doubts) about the potential application.

The insights regarding the numbers of societies people were a part of (average 4) and seeing the majority indicating their initial interest in joining more, indicated the troubles societies have with member retention – with this being somewhat demonstrated by interviews with committee members (Interviews 4 and 10 suggest briefly how our app could improve turn out)

There were however some concerns about notifications being overwhelming. This is consistent with the notification heavy nature of WhatsApp and Instagram – which were noted as being the apps people use most for societies.

The information we received about people's favourite apps (TikTok and Instagram) ultimately reinforced our understanding of the prevalence of doomscrolling (see Interviews 7 and 9), which

we will be taking on board as we progress to the design of our user interface. Equally, the generally speedy response time for members contacting societies will need to be something we enable with our UI.

The most common way people interact with posts is likes (or affirming reactions), across Instagram Posts, Stories and WhatsApp messages – with the posts mainly being text/image, though polls and videos also seem to factor somewhat into preferences.

Calendar posts also seem to be a common way societies outreach to members – supporting our core calendar sync feature. Many expressed support for a guest list, and so we'll ensure our UI integrates this.

The process was not linear, and through carrying out product reviews, we were able to adjust our requirements to reflect on the interesting features which applications such as Instagram offer, which will no doubt factor into our development of the project next semester.

User Requirements

Society Member requirements

(Interview Reference is as follows – Ix/Qy: Question number (y) maps to interview (Ix) in Appendix)

1. Users should be able to log in / register (All – implied by account usage)

2. Users should be able to join multiple societies (All – implied by average society count)

3. Users should be able to contact a societies' committee (All – implied by experience contacting them, see I5/Q5, I6/Q6, I8/Q5, I9/Q5)

4. Users should be able toggle on/off notifications for a societies' feed (I1/Q13, I11/Q8-9, I8/Q12)

5. Users should be able to interact (vote in polls / like / tag (invite) friends to event) with posts (Explicit in all bar I2, I7 and I11, implied in I2/Q10 and I11/Q12)

6. Users should be able to add a single societies' event to their device calendar (I8/Q8)

7. Users should be able to add all a societies' events to their device calendar (I1/Q17, I4/Q25, I10/Q19, I11/Q13)

8. Users should be able to view a guest list (I1/Q11, I2/Q12, I3/Q19, I5/Q17, I8/Q16. I9/14)

9. Users should be able to RSVP to an event (required for above)

10. Users should be able to add society members as their friends (All – implied by deference to use other apps)

    - users should be able to see a list of society members (required for above)

Committee Member requirements – see Committee Interviews 1/4/10/11

11. Users should be able to manage posts (create / modify) (required by nature of system)

    - users should be able to link their post to an event (supports UserReq 9)

12. Users should be able to manage (create / modify) events (required for UserReq 6 and 7)

- switch an event's status (to public / private / scheduled) (implied I10/Q21)

13. Users should be able to respond to a society member (required for UserReq 3)

System Requirements

Concepts

SM – Society Members
CM – Committee Members
MIDS – Multi-item Data Structure
ID – A String
File Path – A String

SM (Society Members) Users have:

An Email [must be a valid port.ac.uk email address | not-null, unique] (String)
A password [minimum 8 characters | not-null] (String)
List of societies [nullable] (MIDS, composed of IDs, with settings)
List of friends [nullable] (MIDS, composed of IDs)

CM (Committee Members) Users have:

An Email [must be a valid upsu.net email address | not-null, unique] (String)
A password [min 8 chars | not-null] (String)
Society ID [not-null] (string)

Societies:

List of posts [nullable] (MIDS, composed of IDs)
List of events [nullable] (MIDS, composed of IDs)
List of members [nullable] (MIDS, composed of IDs)
List of settings [not-null] (MIDS)
Message Inbox (MIDS)

Posts:

List of Likes [default 0 | not-null] (List of IDs of people who have liked the post)
Post Data [not-null] (MIDS, see below)
Linked Event [nullable] (String)
Post Configuration [not-null] (MIDS)

Events:

Post Data [not-null] (MIDS, see below)
Linked Event [nullable] (String)
Event Configuration [not-null] (MIDS)

Post Data:

Description / Text [max 2,200 chars | not-null] (String)
List of Images [nullable] (MIDS, composed of file paths)
Poll [nullable] (MIDS, see below)

Poll:

List of Options [min 2 items | not-null] (MIDS, composed of Strings, with sets of votes (IDs))

Functional requirements (User Requirement Ref)

1. The system should authenticate users if their email and password match in the database (1)

2. The system should check the database to ensure no other account exists if a user attempts to register an email which is already in the database (1)

3. The system should add a societies ID to a user's list of societies upon user confirmation (2)

4. The system should update to demonstrate to the user they've joined a society (2)

5. The system should offer the user a text field to send their query directly to the society (3)

6. The system should alert the society that they have a query in their inbox, and update it with a response (3)

7. The system should update the database with a user's per-society notification settings (4)

8. The system should update the database to reflect a user's like (or removal of), on a post (5)

9. The system should update the database to reflect a user's vote in a poll. (5)

10. The system should demonstrate the present of a user's like of a post (5)

11. The system should demonstrate a user's vote in a poll (5)

12. The system should generate an ICS file for either a single event or a society's upcoming events, through data fetched from the database (6, 7)

13. The system should direct the user to their calendar app of choice to save the event, or save the ICS file should no appropriate app be installed (6, 7)

14. The system should update to demonstrate to the user that the event was added (6, 7)

15. The system should query the database to return a list of attendees for an event (8)

16. The system should update the database with a user's RSVP (Boolean, attending / not) (9)

17. The system should update to demonstrate to the user their RSVP status (9)

18. The system should query the database to return a list of members within a society for users to view (10)

19. The system should verify the target user is accepting friend requests (10)

20. The system should update both users' outgoing/incoming requests, notifying target users of new requests (10)

21. The system should notify a target user that they have been invited to an event (5)

22. The system should verify that all post details are filled with valid data (11)

23. The system should send an API request to the backend with the verified post data (11)

24. The system should then add this post to the database via the database connector. (11)

25. The system should fetch an event ID from the database using the restAPI. (12)

26. The system should fetch a Post object from the database and then send a new request to the database manager inserting the eventURL to the posts row in the database. (11)

27. The system should verify that all event details are filled with valid data. (12)

28. The system should send an API request to the backend with the verified event data. (12)

29. The system should then add the new event record to the database via the database manager. (12)

30. The system should fetch the existing event object from the database using the restAPI to populate the fields. (11)

31. The system should verify that the modified text boxes contain valid data. (11, 12)

32. The system should send an API request to the backend with the updated data and the event ID. (12)

33. The system should update the specific event row in the database via the database manager. (12)

34. The system should send an API request to the backend containing the event ID and the selected status tag. (11)

35. The system should query the database manager to locate the specific event. (12)

36. The system should update the status column for that event in the database to reflect the user's selection. (12)

37. The system should fetch the ID of the raised concern from the database using the restAPI. (13)

38. The system should verify that the response text box is filled with valid data. (13)

39. The system should send an API request to the backend linking the new response to the concern's ID. (13)

40. The system should insert the response text into the database via the database manager. (13)

Non-Functional requirements

1. The system should not take more than 3 seconds to load the event dashboard.
2. The system should not take more than 3 seconds to complete the upload of a standard text post.
3. The system should be able to fetch and display a specific event's details within 3 seconds of the user's request.
4. The system will have login information saved, allowing logins to be reused (session persistence).
5. API requests should complete within 2-3 seconds under normal load
6. Maximum file size of 10MB per image
7. Users must be authenticated before creating, modifying, or linking posts/events
8. All input data must be sanitized to prevent SQL injection and XSS attacks