MindGuard

Internet Tool and Technology Lab Synopsis- ICT 3266

February 6, 2024

Abstract

MindGuard aims to address mental health challenges through an innovative platform combining a chatbot with predictive analysis, daily content updates via web scraping, and engaging user features. The primary objective is to assess individuals' mental health, provide personalized insights, and foster a supportive community. The platform offers dynamic content, user engagement features, and a feedback system for continuous improvement.

1. Objective

- Create an interactive platform for mental health assessment using a chatbot questionnaire with personalized responses.
- Utilize web scraping techniques for daily updates of relevant mental health articles.
- Showcase a comprehensive understanding of mental health concepts through the analytical capabilities of the chatbot and content curation.
- Evaluate user engagement features (login, like, share, comment, save, and community forum) for their effectiveness on user satisfaction and platform performance.
- Present recommendations for continuous improvements and future research based on user feedback.
- Integrate a community forum to foster a supportive environment for user engagement in discussions, mutual support, and collaborative sharing of experiences and insights.

2. Methodology

Construct a chatbot designed to analyze user responses, assess mental health, and deliver personalized insights. Utilise effective methods for retrieving and providing daily updates on mental health topics. Enhance user interaction through features like login, like, share, comment, and save, and introduce a community forum for discussions. Integrate a dynamic feedback mechanism, continually refining the platform based on user experiences. Additionally, create a supportive community atmosphere by establishing a dedicated space for users to engage in discussions within the community forum, fostering collaborative and enriching conversations about mental health.

3. Outcome

Anticipated outcomes include a user-friendly platform effectively assessing mental health, delivering relevant content, and fostering a supportive community. The project aims to contribute valuable insights into mental health technology and user engagement for future advancements. Continuous feedback and analysis will guide further enhancements.

4. Team Member details

Name	Registration Number	Roll Number	Semester & Branch	Section
Swapnil Jha	210911292	50	VI (IT)	A
Rasagnya Bethi	210911298	52	VI (IT)	A
Srijan Raj	210911410	71	VI (IT)	A

MindGuard

Internet Tool and Technology Lab Design Document- ICT 3266

Marth 12, 2024

1. Introduction

An innovative platform for mental health, MindGuard provides a variety of interactive tools and materials to help people understand and take better care of their mental health. It is thoughtfully designed to assess users' mental health through a scientific yet user-friendly DASS-42 questionnaire, provide personalized insights, and cultivate a supportive community environment. The platform's primary objectives are to facilitate mental health self-assessment, deliver customized content, and enable peer-to-peer support through community engagement. Leveraging cutting-edge technology, MindGuard ensures a responsive and adaptive service, tailored to meet the evolving needs and preferences of its users, with the ultimate goal of empowering individuals on their journey to mental wellness.

Objectives -

- Create an interactive platform for mental health assessment using a chatbot questionnaire with personalized responses.
- Provide a comprehensive mental health self-assessment through the DASS-42 questionnaire.
- Deliver personalized insights and recommendations based on questionnaire results.
- Utilize web scraping techniques for daily updates of relevant mental health articles.
- Foster a supportive community for peer-to-peer support and engagement.
- Maintain a dynamic platform that evolves with user feedback and the latest mental health research.
- Integrate a community forum to foster a supportive environment for user engagement in discussions, mutual support, and collaborative sharing of experiences and insights.

2. Process Model

The development methodology for MindGuard aligns with the Agile Process Model, which approaches the project iteratively and collaboratively, emphasizing flexibility, customer feedback, and continuous improvement. The Agile Process Model divides the development process into short iterations, typically lasting 1 to 4 weeks, known as sprints, delivering a potentially shippable increment of the software. This allows for early and continuous delivery of valuable functionality, enabling MindGuard to respond quickly to changing requirements and evolving user needs.

A key advantage of the Agile Process Model is its emphasis on customer collaboration, which MindGuard prioritizes throughout development to ensure the final product effectively meets user expectations. Embracing the Agile Process Model allows MindGuard to benefit from its iterative nature, facilitating frequent releases and adaptations. This proves particularly beneficial for MindGuard, where requirements may evolve or require refinement over time. Furthermore, the Agile Process Model promotes continuous improvement through regular reflection and adaptation, and MindGuard leverages sprint reviews and retrospectives to identify areas for enhancement and make adjustments accordingly.

In conclusion, the Agile Process Model offers MindGuard a structured framework for iterative and collaborative development. By adopting this methodology, MindGuard can deliver a responsive and user-centric platform that evolves in alignment with user needs and industry trends.

3. Architecture Design

The MindGuard mental health platform is designed with a structured approach, delineating the key components and their interactions.

i. Frontend Layer

- *Technology Stack*: React.js is utilized for crafting a dynamic and responsive user interface.
- *Main Components*:
  - Registration, Login, Logout UI
  - Article browsing, searching, and filtering interface
  - DASS-21 Questionnaire interface
  - Community forum for posting, commenting, and upvoting
  - User dashboard for profile and preferences management
- *Authentication*: Seamlessly integrated with Firebase for Google authentication, handling user sign-in and sign-up processes.

ii. Backend Layer

- *Technology Stack*: Express.js serves as the framework for creating essential API endpoints.

- *Main Components*:
  - User Authentication and Management:
  - Manages user registration, login, and logout processes.
  - Handles session management and user profile updates.
- *Web Scraping Service*:
  - Utilizes Puppeteer to scrape articles from specified mental health websites (e.g., Verywell Mind, HelpGuide, Psych Central).
  - Scheduled intervals ensure the article collection stays updated.
- *Article Management*:
  - Provides APIs to fetch, search, and filter articles stored in the MongoDB database.
- *Questionnaire Service*:
  - Manages the DASS-21 Questionnaire logic and offers an API for submitting responses and receiving predictions on mental health conditions.
- *Community Forum Service*:
  - Facilitates posting, viewing, commenting, and upvoting within the community forum.

iii. Database Layer

- *Technology Stack*: MongoDB is selected for its flexibility in managing document-based data.
- *Collections*:
  - User Data: Stores information about registered users, including authentication details, profile information, and user preferences.
  - Scraped Article Data: Contains articles scraped from external websites, including metadata such as titles, summaries, URLs, and categories.
  - Community Forum Data: Stores posts, comments, and upvotes related to stories shared within the community forum.

Overall Architecture Flow:

1. Users interact with the frontend application for various activities, such as browsing articles, participating in the community forum, or completing the DASS-21 questionnaire.

2. The frontend communicates with the backend through API calls to fetch data or perform operations (e.g., user authentication, article retrieval, questionnaire submission).

3. The backend performs the necessary logic, accesses the MongoDB database for data storage/retrieval, and uses Puppeteer for web scraping tasks.

4. External services like Firebase and the targeted mental health websites integrate seamlessly with the backend to provide authentication and content, respectively.

4. Use Case Diagram

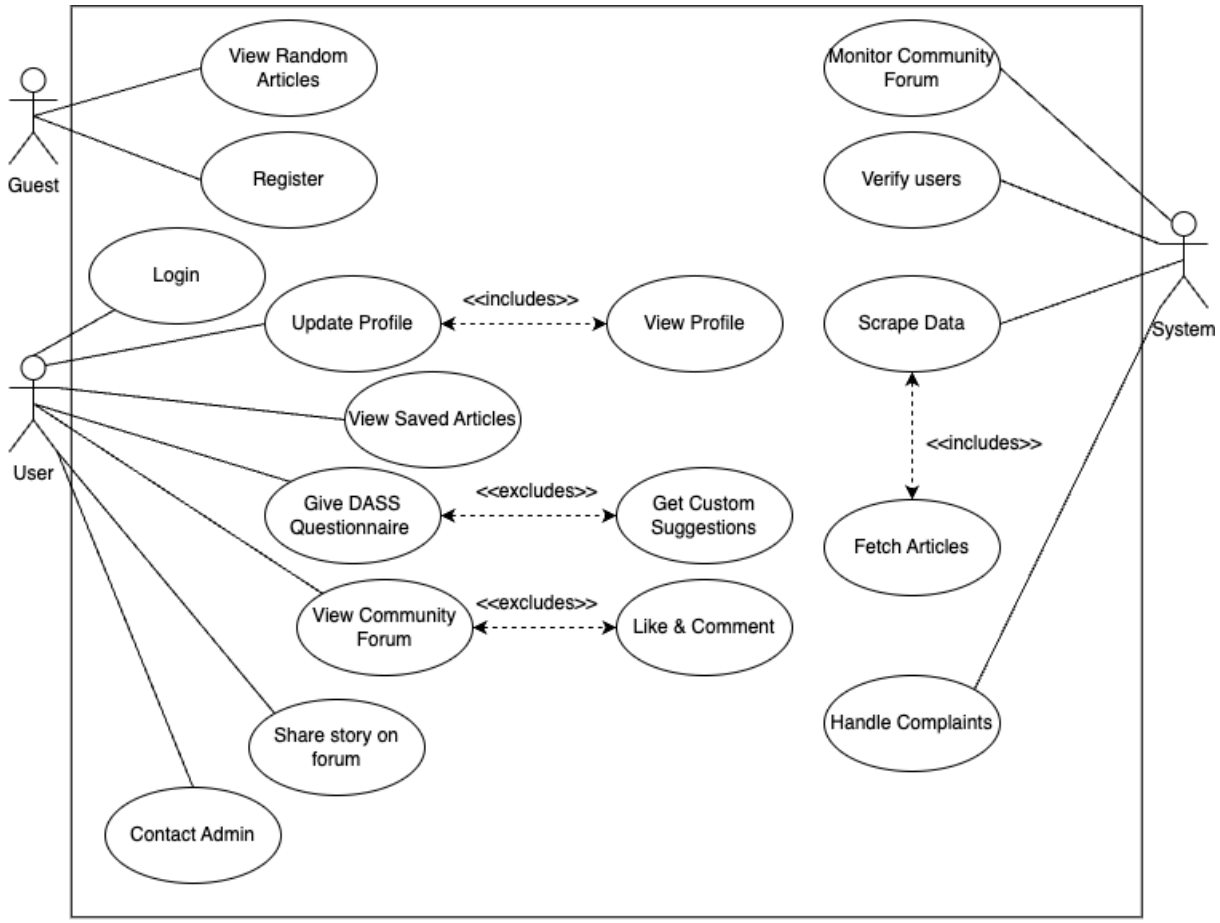


Figure 1: Use Case Diagram

4.1. Use Case Diagram Description

1. Actors:

- Guest: This is a user without an account or someone who is not logged in.
- User: This is a registered user who has more privileges than a guest, such as saving articles or participating in community forums.
- System: Represents automated processes or system-level operations that don't require direct human input.

2. Use Cases:

- *For Guest*:
  - View Random Articles: The ability to view articles without the need to register or log in.
  - Register: The option for a guest to create a new user account.

- *For User*:
  - Login: Authenticate to access additional features.
  - Update Profile: Change or add information to their profile.
  -View Profile: Look at their own or others' user profile.
  - View Saved Articles: Access previously saved articles.
  - Give DASS Questionnaire: Complete a questionnaire (presumably for mental health assessment as DASS refers to Depression, Anxiety, and Stress Scales).
  - Get Custom Suggestions: Receive article recommendations based on user preferences or questionnaire results.
  - View Community Forum: Engage with the community forum.
  - Like & Comment: Interact with content by liking or commenting on it.
  - Share Story on Forum: Post personal stories or experiences on the forum.
  - Contact Admin: Reach out to the system administrator for help or to report issues.

- *For System*:
  - Monitor Community Forum: Automatically or manually oversee forum activity.
  - Verify Users: Confirm the authenticity of users.
  - Scrape Data: Collect data from various sources for use within the system.
  - Fetch Articles: Retrieve articles from a database or external source to be presented to the user.
  - Handle Complaints: Manage and respond to user complaints.

3. Relationships and Dependencies:

i. *Include Relationships (<<include>>)*:
- "Update Profile" includes "View Profile". This means that whenever the "Update Profile" use case is executed, "View Profile" is also necessarily performed as part of the process.
- "Scrape Data" includes "Fetch Articles". The system's process of scraping data includes the specific activity of fetching articles.

ii. *Exclude Relationships (<<exclude>>)*:

- "Give DASS Questionnaire" excludes "Get Custom Suggestions". When the "Give DASS Questionnaire" use case is executed, it excludes the use case "Get Custom Suggestions", indicating that these two actions cannot occur simultaneously.
- "View Community Forum" excludes "Like & Comment". This suggests that users cannot like or comment while they are simply viewing the forum, indicating a separate phase or mode of interaction.

5. Activity Diagram

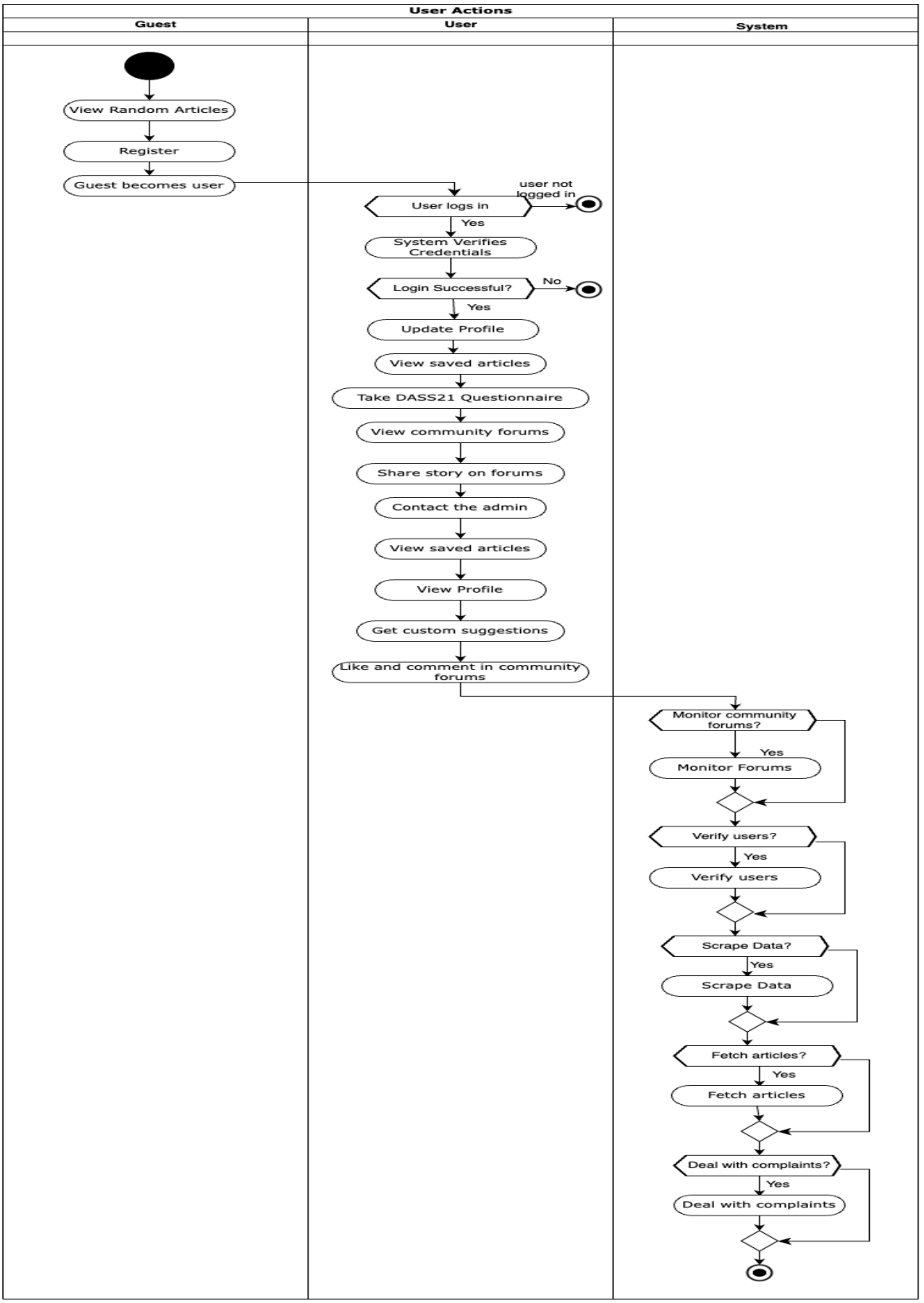


Figure 2: Activity Diagram

5.1. Activity Diagram Description

1. For Guests:

- Starts with "View Random Articles".
- Then "Register".
- After registration, the guest becomes a user.

2. For Users:

- The user begins with the decision point of logging in.
- If the user is not logged in, the flow ends there.
- If the user logs in, the system verifies credentials.
- If login is successful, the user proceeds to "Update Profile".
- After updating the profile, the user can "View Saved Articles".
- The user can then "Take DASS21 Questionnaire".
- Following that, the user can "View Community Forums".
- They can "Share Story on Forums".
- The user might need to "Contact the Admin".
- The user can revisit "View Saved Articles".
- Subsequently, the user can "View Profile".
- Based on the profile, they can "Get Custom Suggestions".
- Finally, the user can engage by choosing to "Like and Comment in Community Forums".

3. For the System:

- The system has a series of decision points where it can perform certain actions based on conditions:
- "Monitor Community Forums" if the condition is met.
- "Verify Users" if required.
- "Scrape Data" if needed.
- "Fetch Articles" when necessary.
- "Deal with Complaints" as they arise.

4. Flow Finalization:

- The system's flow ends after it performs any of the above actions based on the decision points.

6. Sequence Diagram

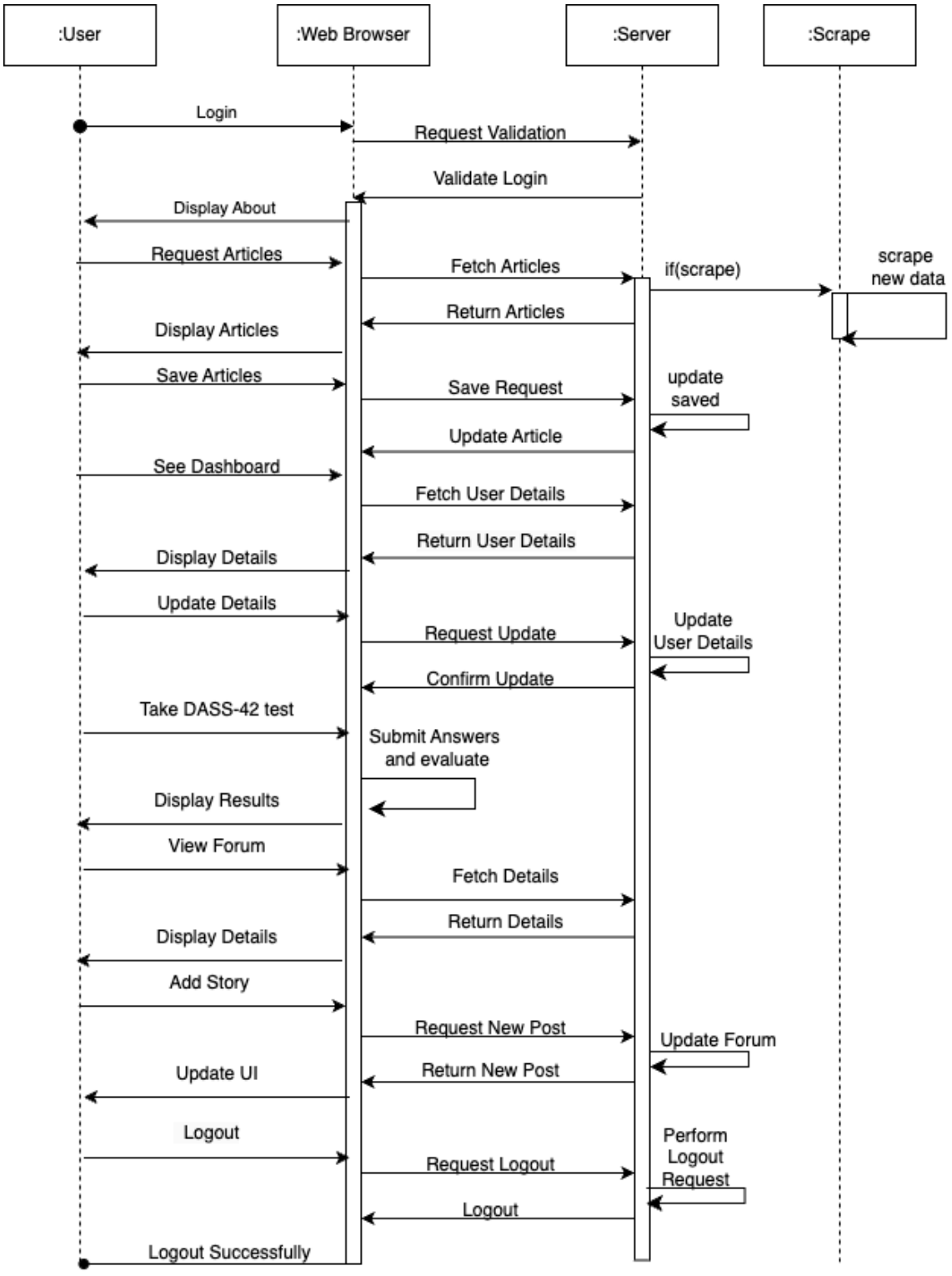


Figure 3: Sequence Diagram

6.1. Sequence Diagram Description

Login and Article Retrieval: The User (U) initiates the sequence by logging into the system through the Web Browser (WB). The Web Browser requests login validation from the Server (S), which validates the credentials and then prompts the Web Browser to display an introductory page.

After successful login, the User requests articles. The Web Browser communicates with the Server to fetch articles, and in the case that new data is needed, the Server has a conditional interaction with a Scrape module (:Scrape) to scrape new data.

Article Interaction and Dashboard Usage: The Server returns the articles to the Web Browser, which displays them to the User. The User selects articles to save, and the Server processes this request and updates the saved articles.

Subsequently, the User decides to view their dashboard, prompting the Server to fetch and return the user's details, which are displayed by the Web Browser.

User Profile Updates and DASS-42 Test: If the User updates their details, the Web Browser sends this request to the Server, which updates the details and confirms the update. The User then takes the DASS-42 test, with the Web Browser submitting the answers to the Server for evaluation. The results are displayed back to the User.

Forum Participation and Story Sharing: The User accesses the community forum through the Web Browser. The Server fetches the forum details and returns them for display. The User contributes by adding a story to the forum, and the Server updates the forum content and confirms the new post.

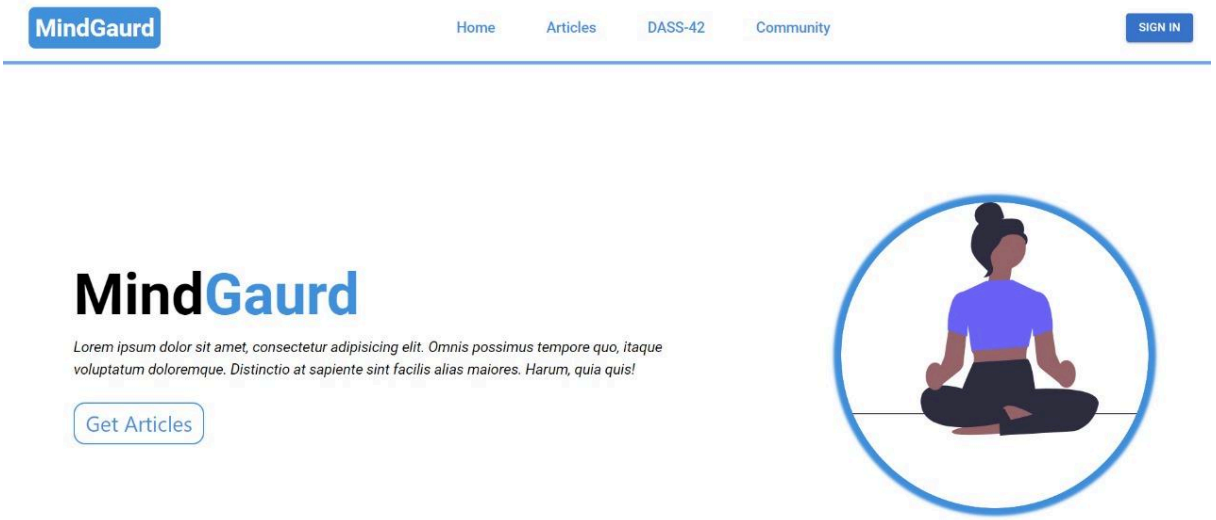
Logout Procedure: Finally, the User logs out through the Web Browser, which sends a logout request to the Server. The Server processes the logout and ensures that the User is logged out successfully.

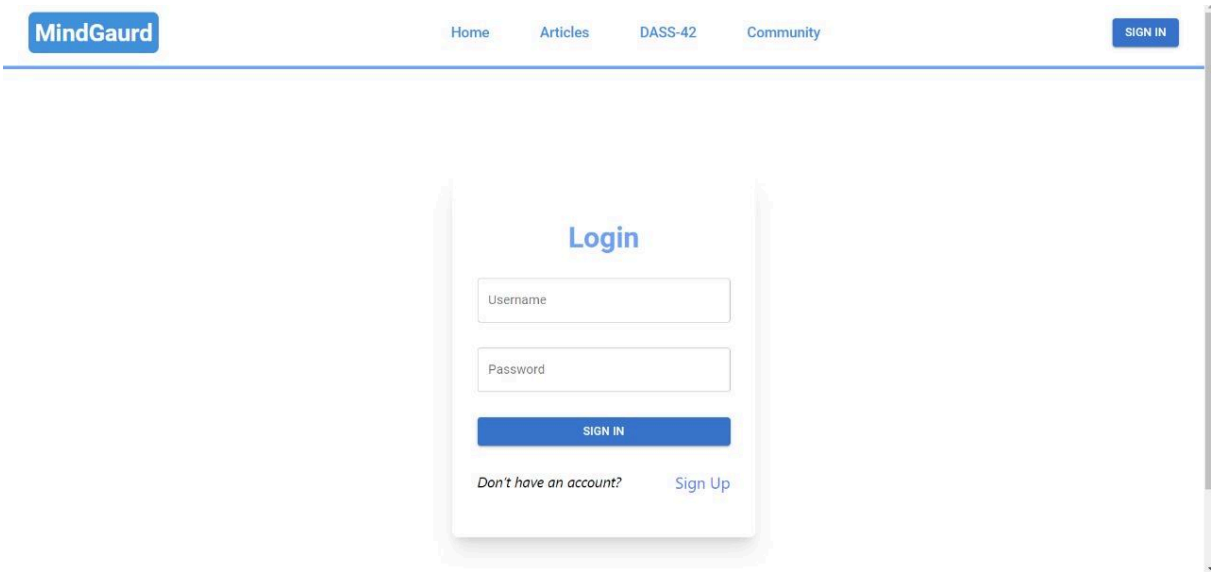
Sequence Conclusion: The sequence concludes with the User successfully logging out of the system, which is processed by the Server. This sequence diagram details the flow of interactions between the User, Web Browser, Server, and Scrape module, depicting the step-by-step activities and communications that lead to a comprehensive user experience from logging in to sharing content on the forum.

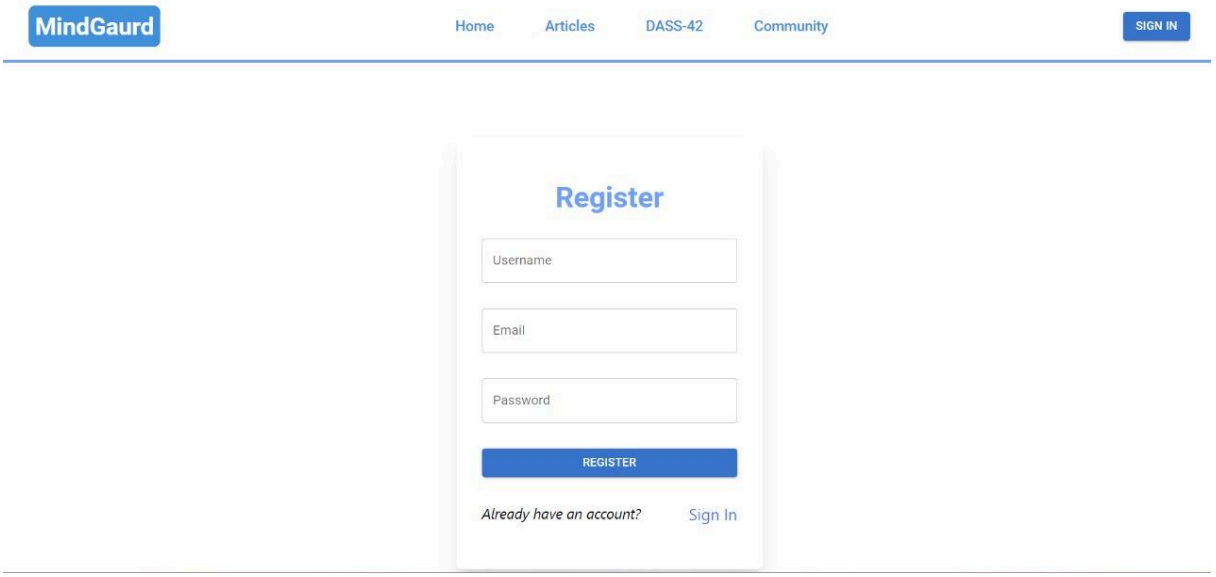
7. Front-end UI Design

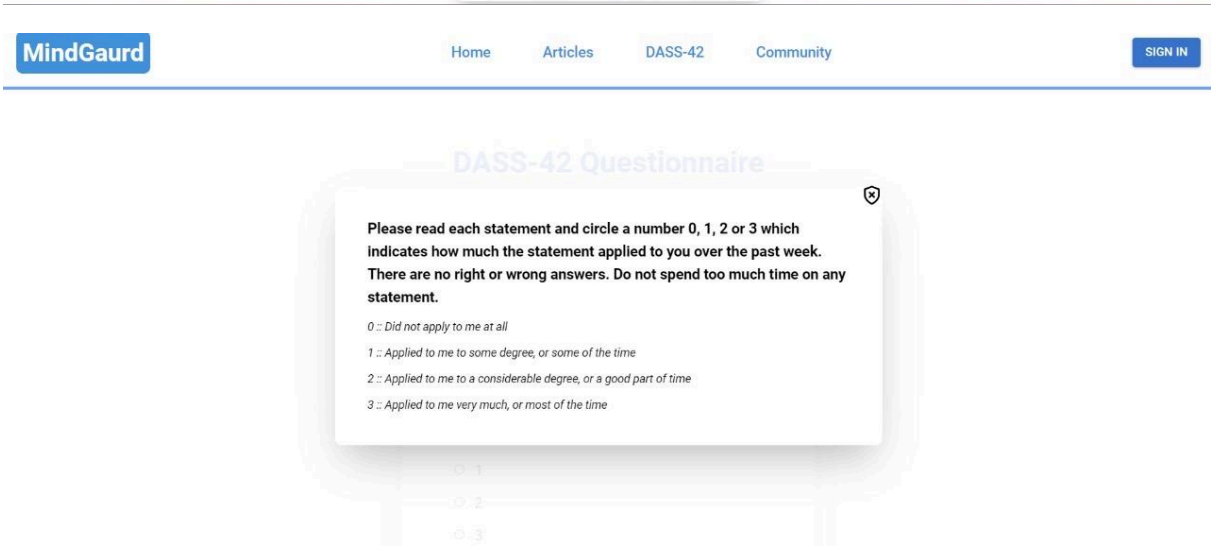
Our project, MindGuard, showcases a clean and inviting interface, with a navigation bar that's easy to navigate, guiding users through the different sections like Articles and Community, alongside a straightforward Sign In option. The registration is uncomplicated,

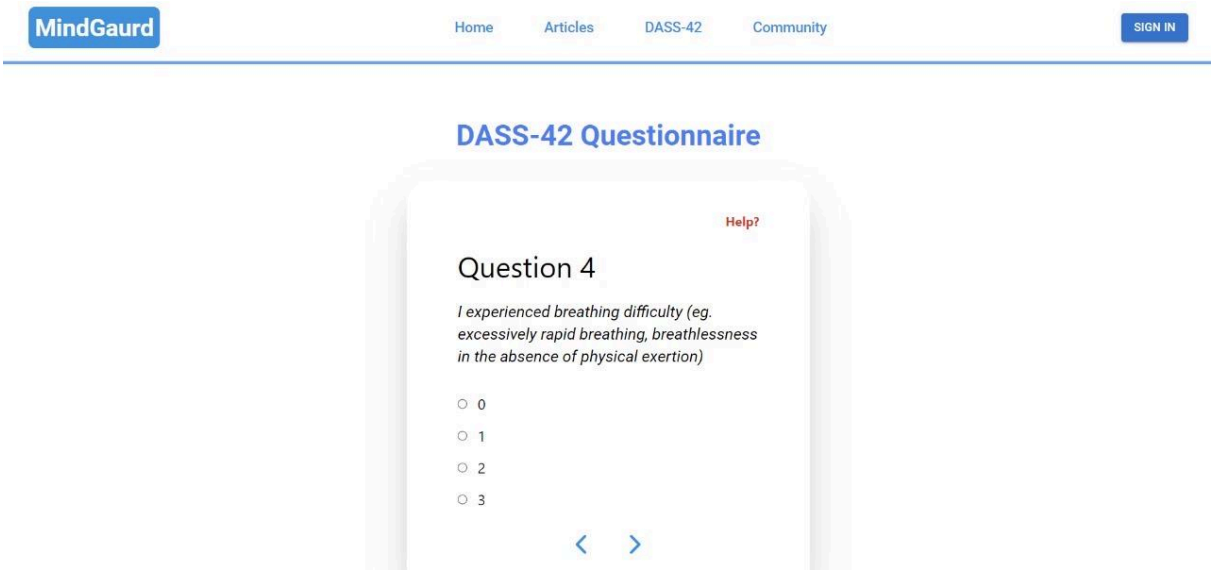
encouraging quick user engagement. Articles are presented in an attractive card layout that invites interaction with likes and comments, enhancing user engagement. Our DASS-42 questionnaire is designed for user comfort, minimizing distractions to aid in the reflective process of self-assessment. The overall design is intentionally minimalist, emphasizing usability and a stress-free experience.

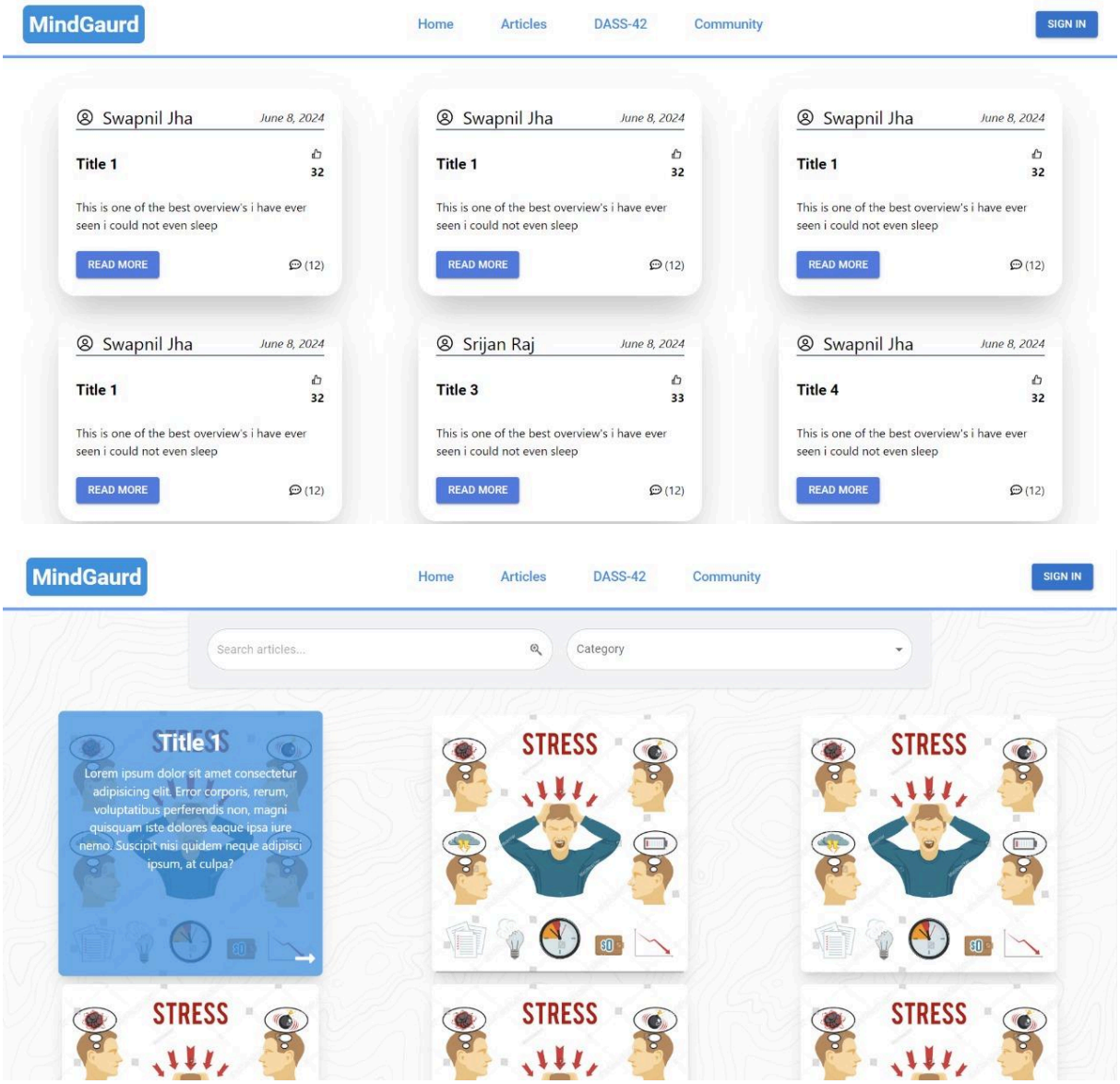





MindGaurd
Home
Articles
DASS-42
Community
SIGN IN
Register
Username
Email
Password
REGISTER
Already have an account?
Sign In


MindGaurd
Home
Articles
DASS-42
Community
SIGN IN
DASS-42 Questionnaire
Please read each statement and circle a number 0, 1, 2 or 3 which indicates how much the statement applied to you over the past week. There are no right or wrong answers. Do not spend too much time on any statement.
0 :: Did not apply to me at all
1 :: Applied to me to some degree, or some of the time
2 :: Applied to me to a considerable degree, or a good part of time
3 :: Applied to me very much, or most of the time


MindGaurd
Home
Articles
DASS-42
Community
SIGN IN
DASS-42 Questionnaire
Help?
Question 4
I experienced breathing difficulty (eg. excessively rapid breathing, breathlessness in the absence of physical exertion)
0
1
2
3



8. Conclusion

In conclusion, the MindGuard project stands as a testament to our dedication to mental health awareness and support. It is meticulously designed to serve as a comprehensive digital companion for individuals seeking to enhance their mental well-being. By leveraging an iterative Agile development process, the platform ensures a responsive and evolving interface that aligns closely with the user's journey—from the initial mental health assessment through the DASS-42 questionnaire to continuous engagement within a supportive community.

MindGuard's design document reflects our attention to user-centricity, offering a seamless and intuitive experience for users. Key features such as personalized insights, a rich

repository of articles, and community-driven support are central to our commitment to providing users with reliable and empowering tools for mental health management.

Developing a platform that not only meets but surpasses user expectations and cultivates a culture of growth, learning, and sharing is the main objective. MindGuard's architecture is built for scalability and adaptation, making it ideally positioned to help users as they traverse their mental health journeys with a system that respects their privacy and personal development. In addition to setting a new benchmark for digital mental health resources, this initiative highlights the important role that technology can play in the field of mental health and demonstrates our capacity to combine technological innovation with caring care. We are ready to provide a revolutionary experience that promotes user involvement, contentment, and general improvements to mental health.

9. Team Member details

Name	Registration Number	Roll Number	Semester & Branch	Section
Swapnil Jha	210911292	50	VI (IT)	A
Rasagnya Bethi	210911298	52	VI (IT)	A
Srijan Raj	210911410	71	VI (IT)	A