1. User Authentication

Description: This feature allows users to log in to the system securely using their credentials.

```
Algorithm: User Authentication
Input: username, password
Output: Authentication status (success/failure)

1. Begin
2. Prompt user to enter username and password
3. Retrieve stored hashed password from database for the given username
4. If username exists:
       a. Hash the input password using the same hashing algorithm
       b. Compare the hashed input password with the stored hashed password
       c. If they match:
              i. Grant access to the system
              ii. Return "Authentication successful"
       d. Else:
              i. Return "Authentication failed: Incorrect password"
5. Else:
       a. Return "Authentication failed: User not found"
6. End6. End
```

2. Profile Management

Description: This feature allows users to manage their personal information and preferences.

```
Algorithm: Profile Management
Input: userID, updatedProfileData
Output: Updated profile status (success/failure)

1. Begin
2. Retrieve user profile using userID
3. If profile exists:
       a. Update profile fields with updatedProfileData
       b. Save changes to the database
       c. Return "Profile updated successfully"
4. Else:
       a. Return "Profile update failed: User not found"
5. End
```

3. View Events

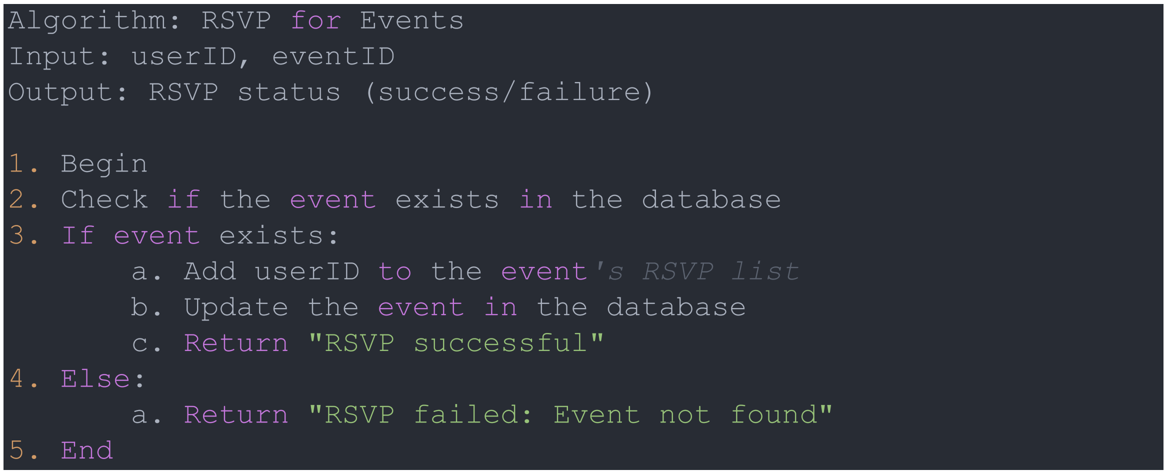
Description: This feature allows users to view a list of academic events.

```
Algorithm: View Events
Input: userID
Output: List of events

1. Begin
2. Retrieve events from the database
3. Filter events based on user preferences (if any)
4. Display events to the user
5. End
```

4. RSVP for Events

Description: This feature allows users to RSVP for events they wish to attend.

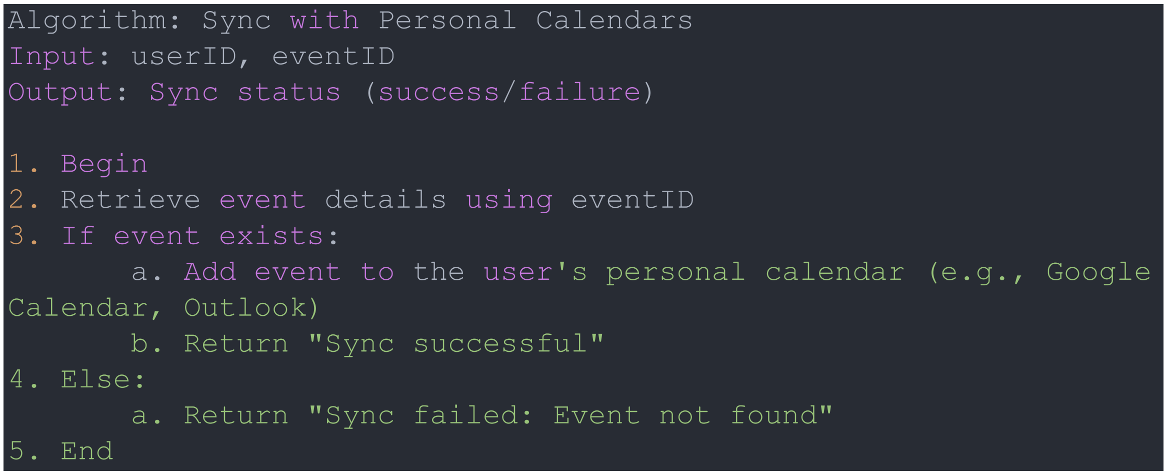


```
Algorithm: RSVP for Events
Input: userID, eventID
Output: RSVP status (success/failure)

1. Begin
2. Check if the event exists in the database
3. If event exists:
       a. Add userID to the event's RSVP list
       b. Update the event in the database
       c. Return "RSVP successful"
4. Else:
       a. Return "RSVP failed: Event not found"
5. End
```

5. Sync with Personal Calendars

Description: This feature allows users to sync event dates with their personal calendars.



```
Algorithm: Sync with Personal Calendars
Input: userID, eventID
Output: Sync status (success/failure)

1. Begin
2. Retrieve event details using eventID
3. If event exists:
       a. Add event to the user's personal calendar (e.g., Google
Calendar, Outlook)
       b. Return "Sync successful"
4. Else:
       a. Return "Sync failed: Event not found"
5. End
```

6. AI-Suggested Events

Description: This feature suggests events to users based on their interests and past behavior.

```
Algorithm: AI-Suggested Events
Input: userID
Output: List of suggested events

1. Begin
2. Retrieve user preferences and past event attendance from the database
3. Use AI model to predict events of interest based on user data
4. Return list of suggested events
5. End
```

7. Resource Sharing

Description: This feature allows users to upload and share academic resources.

```
Algorithm: Resource Sharing
Input: userID, resourceFile, category
Output: Resource upload status (success/failure)

1. Begin
2. Upload resourceFile to the server
3. Store resource metadata (e.g., userID, category, timestamp) in the database
4. Return "Resource uploaded successfully"
5. End
```

8. Categorized Content Upload

Description: This feature allows users to upload content under specific categories.

```
Algorithm: Categorized Content Upload
Input: userID, contentFile, category
Output: Content upload status (success/failure)

1. Begin
2. Upload contentFile to the server
3. Store content metadata (e.g., userID, category, timestamp) in the database
4. Return "Content uploaded successfully"
5. End
```

9. Participate in Discussions

Description: This feature allows users to participate in community discussions.

```
Algorithm: Participate in Discussions
Input: userID, discussionID, message
Output: Discussion participation status (success/failure)

1. Begin
2. Retrieve discussion using discussionID
3. If discussion exists:
       a. Add message to the discussion
       b. Update discussion in the database
       c. Return "Message posted successfully"
4. Else:
       a. Return "Discussion not found"
5. End
```

10. AI-Powered Chatbot (FAQ)

Description: This feature provides automated responses to common institutional queries.

```
Algorithm: AI-Powered Chatbot
Input: userQuery
Output: Chatbot response

1. Begin
2. Analyze userQuery using NLP
3. Retrieve the most relevant response from the FAQ database
4. Return response to the user
5. End
```

11. NLP-Based Suggestions

Description: This feature provides personalized responses to complex queries using NLP.

```
Algorithm: NLP-Based Suggestions
Input: userQuery
Output: Personalized response

1. Begin
2. Analyze userQuery using advanced NLP techniques
3. Generate a personalized response based on user context and
history
4. Return personalized response to the user
5. End
```

12. Recommendation Engine

Description: This feature provides personalized content recommendations to users.

```
Algorithm: Recommendation Engine
Input: userID
Output: List of recommended content

1. Begin
2. Retrieve user preferences and past interactions from the database
3. Use recommendation algorithm (e.g., collaborative filtering) to generate content suggestions
4. Return list of recommended content
5. End
```

13. Post Suggestions

Description: This feature suggests posts to users based on their interests.

```
Algorithm: Post Suggestions
Input: userID
Output: List of suggested posts

1. Begin
2. Retrieve user preferences and past interactions from the database
3. Use AI model to predict posts of interest based on user data
4. Return list of suggested posts
5. End
```

14. Mentorship Matching

Description: This feature matches students with mentors based on their interests and goals.

```
Algorithm: Mentorship Matching
Input: userID
Output: Matched mentor

1. Begin
2. Retrieve user profile and preferences from the database
3. Use matching algorithm to find the best mentor based on user
data
4. Return matched mentor
5. End
```

15. Direct Messaging

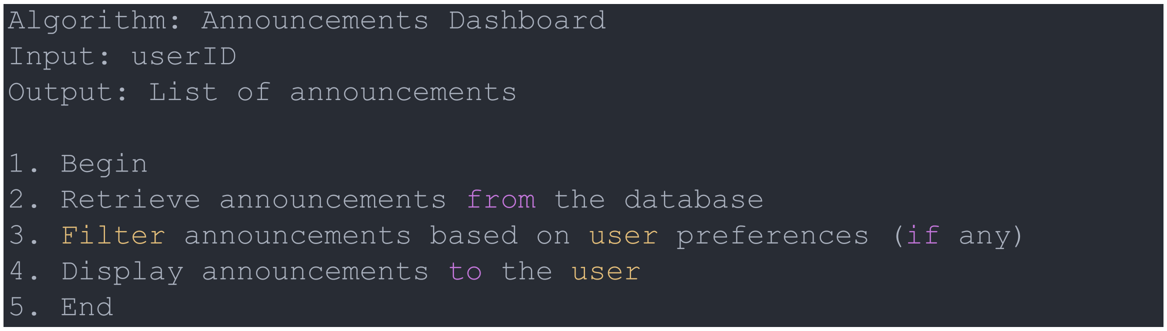
Description: This feature allows users to send direct messages to mentors or peers.

```
Algorithm: Direct Messaging
Input: senderID, receiverID, message
Output: Message send status (success/failure)

1. Begin
2. Store message in the database with senderID, receiverID, and
timestamp
3. Notify receiver of the new message
4. Return "Message sent successfully"
5. End
```

16. Announcements Dashboard

Description: This feature displays important announcements and updates to users.

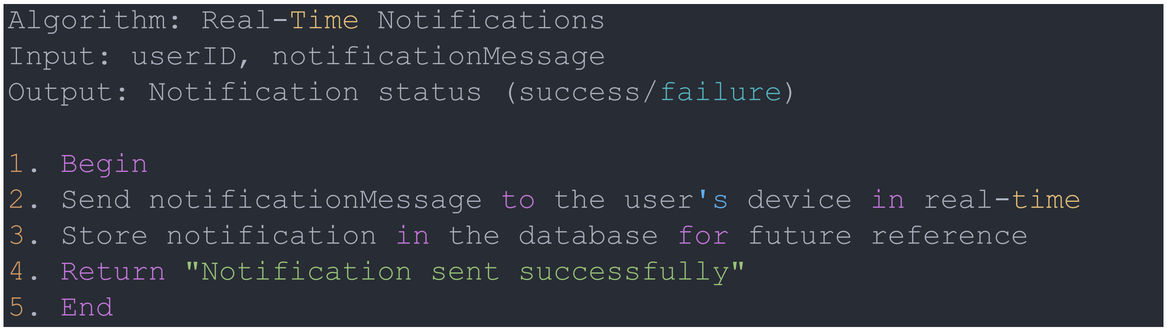


```
Algorithm: Announcements Dashboard
Input: userID
Output: List of announcements

1. Begin
2. Retrieve announcements from the database
3. Filter announcements based on user preferences (if any)
4. Display announcements to the user
5. End
```

17. Real-Time Notifications

Description: This feature sends real-time notifications to users about updates or events.



```
Algorithm: Real-Time Notifications
Input: userID, notificationMessage
Output: Notification status (success/failure)

1. Begin
2. Send notificationMessage to the user's device in real-time
3. Store notification in the database for future reference
4. Return "Notification sent successfully"
5. End
```