SW Engineering CSC648/848
Section-04 Fall 2024
Team 01

Project Title :

“ChillMate”

Eiffel Valentino (Team Leader)
Luis Carrillo (GitHub Master)
Gio Jung (Scrum Master)
Chun Kai Liu (FrontEnd Leader)
William Widjaja (BackEnd Leader)
Sneha katturu (FrontEnd)
Jay Lodha (BackEnd)

Milestone 2

Date Submitted	
Date Revised	

1. Data Definitions V2

Old Version

User - entity that stores the information of registered accounts
- firstName: First name of user
- lastName: Last name of user
- email: Email address of user. Used for login
- password: Password of user. Used for login, and will be hashed
- Address: home address of user
- phone number: phone number of user

Chat Interaction - the dialog exchanged between the student and the chatbot.
- message: Content provided by the user
- response: Content provided by the chatbot
- timestamp: Time of the interaction
- sentimentAnalysis: A score or indicator representing the emotional tone of the student's message

Resources - entity that provide resource to users
- Recourse Title : Title of the resource
- Recourse Link : Linke of the resource
- Recourse Body : Details of the resource
- SFSU Specific (Yes or NO) : Is the resource SFSU Specific?

Mental Health Entry - entity that stores the user current health and emotional status
- mood: The current mood of the user (e.g., Happy, Sad, Anxious, Stressed)
- timestamp: The time the entry was made
- mentalHealthScore: A numerical or categorical score based on the student's inputs (e.g., stress or anxiety levels)

Goal/Task - entity that stores the user goal / tasks that they want to achieve
- Goal: the goal that user input
- User: which user wants to achieve this goal
- Type: whether the goal will be recurring or just one time
- Status: tracks whether the goal have been accomplished or not

Journal Entry - entity that stores the journal entry of the user
- content: journal entry based on the user input

- timestamp: Time the journal entry was created
- sentimentTag: Optional sentiment analysis of the content to assess emotional undertones

Reminder/Notification - alert sent to the user to remind them to interact with the app or prompt a wellness check-in

- type: Can be a reminder (e.g., for mood check-in), or an alert (e.g., if mental health score indicates high stress)
- timestamp: When the notification was sent
- trigger: The condition that caused the notification (e.g., absence of interaction for a certain period)

Data and Analytics - data that has been stripped of personally identifiable information, used for analytics and reporting. Also to monitor app performance and review aggregate mental health data

- moodTrends: Aggregated data without identifiable information
- usageMetrics: General usage patterns like frequency of app interaction
- usageAnalytics: Metrics about app usage

Forum - entity that will store the posts and comments that will be created in the forum

- Text: contains the user input to the forum
- Topic: determines what topic this is about, whether advice, tips, question etc
- Type: determines whether the user input is a comment or a post

Revised Version

User - Collection that stores the information of registered accounts

- SFStateID
- FirstName
- LastName
- Email
- Password
- Address
- PhoneNum
- Age
- Occupation
- EmergencyContactEmail

Chat Interaction - Collection storing dialog exchanged between the student and the chatbot.

- SFStateID
- Message

- Response
- timestamp

Resources - Collection that stores campus specific information
- RecourseTitle
- RecourseLink
- RecourseBody
- ResourceEmbedding

Mental Health Entry - Collection that stores the user current health and emotional status
- SFStateID
- Mood: The current mood of the user (e.g., Happy, Sad, Anxious, Stressed)
- TimeStamp: The time the entry was made

Goal/Task - Collection that stores the user goal / tasks that they want to achieve
- SFStateID
- Goal: the goal that user input
- SubTasks: Array
- Status: tracks whether the goal have been accomplished or not

Journal - entity that stores the journal entry of the user
- SFStateId
- Content: journal entry based on the user input
- timestamp: Time the journal entry was created

Forum - Collection that will store the posts and comments that will be created in the forum
- SFStateID
- Topic: determines what topic this is about, whether advice, tips, question etc
- Text: contains the user input to the forum
- NoofLikes

2. Functional Requirements V2

Old Version

- Creating an account, and being able to log in to the account
- Users have their own profile
- Users can have a conversation with chatbot wherein the chatbot supports features such as:

1. Resource Recommendation: The chatbot could provide personalized resources like articles, videos, or links to campus services based on the user's issue, such as study tips, mental health resources, or academic support.
2. Goal Setting & Tracking: The bot can help students set goals (e.g., academic, fitness, personal) and track progress. It could give friendly reminders and motivation to encourage goal completion.
3. Crisis Response and Escalation: If the chatbot detects signs of distress or crisis, it should be able to escalate the situation by directing the user to a counselor, or provide emergency contacts. We will basically track the user's mood/ tone during the conversation.
    a. Users will have the mood tracker on their own profile that shows as an emoji (P3)
4. Conversational Context Retention: Store conversation context so the chatbot remembers past interactions, allowing it to provide better continuity and personalized suggestions in future sessions.
5. Campus-Specific Information: Offer location-specific details like event reminders, campus maps, office hours for different departments, or FAQs about student services. This can make the bot even more valuable for college students.
6. Survey and Feedback Collection: Periodically ask students about their satisfaction with various aspects of campus life, allowing the institution to gather data on student well-being and experiences
7. Conversational Mood Journal: Allow students to record how they're feeling each day in a conversational style. The chatbot can track this over time and give them insights or encourage them when it notices patterns of stress.

- Provide a journal area where user can use write their problem
- Provide resources for user problems
    - Users should be able to save the resources for future uses
- Give a daily word of encouragement
- A forum where user can post their problems, tips, solutions
    - User can also comment on each other’s post
    - A report feature is also provided to report inappropriate posts
- Users can create a Task list on the application based on the checklist provided by the chat bot (as part of the Goal Setting & Tracking feature from chatbot).

Revised Version

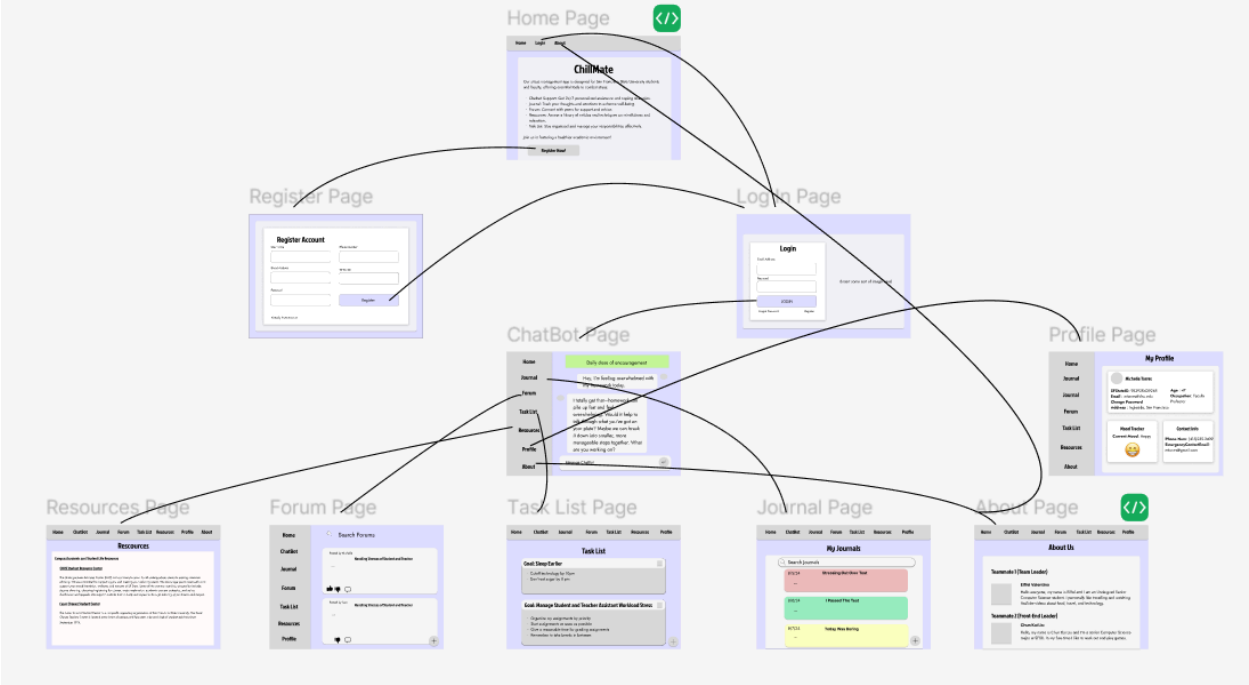
- **Logged in User**
    - Users are able to access the Login Page. (P1)
        - Users are able to login to the website. (P1)
    - Users are able to log out only when they are logged in. (P1)

- Users are able to access Profile Page. (P1)
- Users are able to access the About Page. (P3)
- Users are able to access the Resource Page. (P2)
- Users are able to access the Forum Page. (P1)
    - Users are able to click the “Like” button on the other users’ Forum/Post. (P2)
    - Users are able to comment on the other Users’ Forum/Post (P3)
    - Users are able to post the Forum/Post that will be shared with other Users. (P1)
    - Users are able to click the “report” button on the other users’ Forum/Post (P3)
- Users are able to access the Chatbot Page. (P1)
    - Users can ask the bot to fetch campus specific information based on their input query. (P1)
    - Users can ask the bot to create sub tasks for the goal they are trying to achieve. (P1)
    - Users can have a general conversation with the bot i.e. get general advice. (P2)
- Users are able to access the TaskList Page. (P2)
    - Users are able to create a TaskList by clicking a plus button. (P2)
- Users are able to access the Journal Page. (P1)
    - Users are able to create a Journal by clicking a plus button. (P1)

- **Not Logged in User**
    - Users are able to access the Login Page. (P1)
    - Users are able to access the Registration Page. (P1)
        - Users are able to create an account. (P1)
            - Users must be able to input their First Name. (P1)
            - Users must be able to input their Last Name. (P1)
            - Users must be able to input their SFSU Email. (P1)
            - Users must be able to input their SFSU ID. (P1)
            - Users must be able to input their Password. (P1)
            - Users must be able to input their Address. (P1)
            - Users must be able to input their Emergency Contact. (P1)
            - Users are able to input their age. (P2)
            - Users are able to input their occupation. (P2)
    - Users are able to access About Page. (P3)
    - Users are able to access the Resource Page. (P2)

- **Mood Tracker** (Pre defined set of moods: Happy, Sad, Emotional, Stresses, Anxious, Neutral, etc)
    - User Mood will be derived based on the context in the journal page. (P1)
    - Users will be able to view the tracked mood in the Profile section. (P1)
    - Mood tracker will trigger an email notification to the emergency contact email mentioned by the user while registration if we analyze a constant dip in user's mood. For example, if we notice that the user's mood has been tagged as "Stressed" or "Anxious" for 3-4 days in a row, then the emergency contact will be notified. (P1)

3. UI Mockups and UX Flows(Using Figma)



We went ahead and added color because we feel color is an important visual aspect for the user to feel welcome and at ease right when they enter our website.

4. High Level Architecture, Database Organization

APIs

1. Login Validation (/login)
    a. API to check whether credential that is used in login is correct
    b. API will return error message when input is not correct
2. Signup Validation (/register)
    a. API will dump the user input to the collection
    b. If the user input is already available, then API will return error message saying that the user is already in the collection

3. Chatbot specific endpoints-
    a. API to fetch campus specific information based on user query (/chatbot/campus_specific_info)
    b. API to create sub tasks for goal inputted by user (/chatbot/goal)
    c. API to have a general conversation with chatbot (/chatbot/general)
    d. API to derive user mood based on information present in journal (/chatbot/mood_tracker)
4. Forum Transactions\
    a. API to fetch all data from forum collection to show in frontend
    b. API will collect data from user input and put it in the forum collection

DB Organization

Collections:

1. **User:** {

```
"SFStateID": {
  "$type": "int",
  "description": "9-digit number"
},
"FirstName": {
  "$type": "string"
},
"LastName": {
  "$type": "string"
},
"Email": {
  "$type": "string",
  "description": "Valid email format"
},
"Password": {
  "$type": "string",
  "description": "Stored hashed password"
},
"Address": {
  "$type": "string"
},
"PhoneNum": {
  "$type": "string",
  "description": "Phone number in string format"
},
"Age": {
  "$type": "int"
},
"Occupation": {
```

```
    "$type": "string"
   },
   "EmergencyContactEmail": {
    "$type": "string",
    "description": "Valid email format"
   }
  }
```

2. **ChatInteraction**: {

```
   "SFStateID": {
    "$type": "int",
    "description": "9-digit number (same as the User's SFStateID)"
   },
   "Message": {
    "$type": "string"
   },
   "Response": {
    "$type": "string"
   },
   "Timestamp": {
    "$type": "date",
    "description": "Timestamp of the interaction"
   }
  }
```

3. **Recourses**: {

```
   "RecourseTitle": {
    "$type": "string"
   },
   "RecourseLink": {
    "$type": "string",
    "description": "Valid URL format"
   },
   "RecourseBody": {
    "$type": "string"
   },
   "ResourceEmbedding": {
    "$type": "array",
    "description": "Array of embedding values, likely numbers (floats or integers)"
   }
  }
```

4. **MentalHealthEntry**: {
   "SFStateID": {
    "$type": "int",
    "description": "9-digit number (same as the User's SFStateID)"
   },
   "Mood": {
    "$type": "string",
    "description": "The user's current mood, e.g., Happy, Sad, Anxious, Stressed"
   },
   "Timestamp": {
    "$type": "date",
    "description": "Timestamp of the mood entry"
   }
  }

5. **GoalTask**: {
   "SFStateID": {
    "$type": "int",
    "description": "9-digit number (same as the User's SFStateID)"
   },
   "Goal": {
    "$type": "string",
    "description": "The user's main goal"
   },
   "SubTasks": {
    "$type": "array",
    "description": "Array of strings representing sub-tasks"
   },
   "Status": {
    "$type": "bool",
    "description": "True if the goal is accomplished, false otherwise"
   }
  }

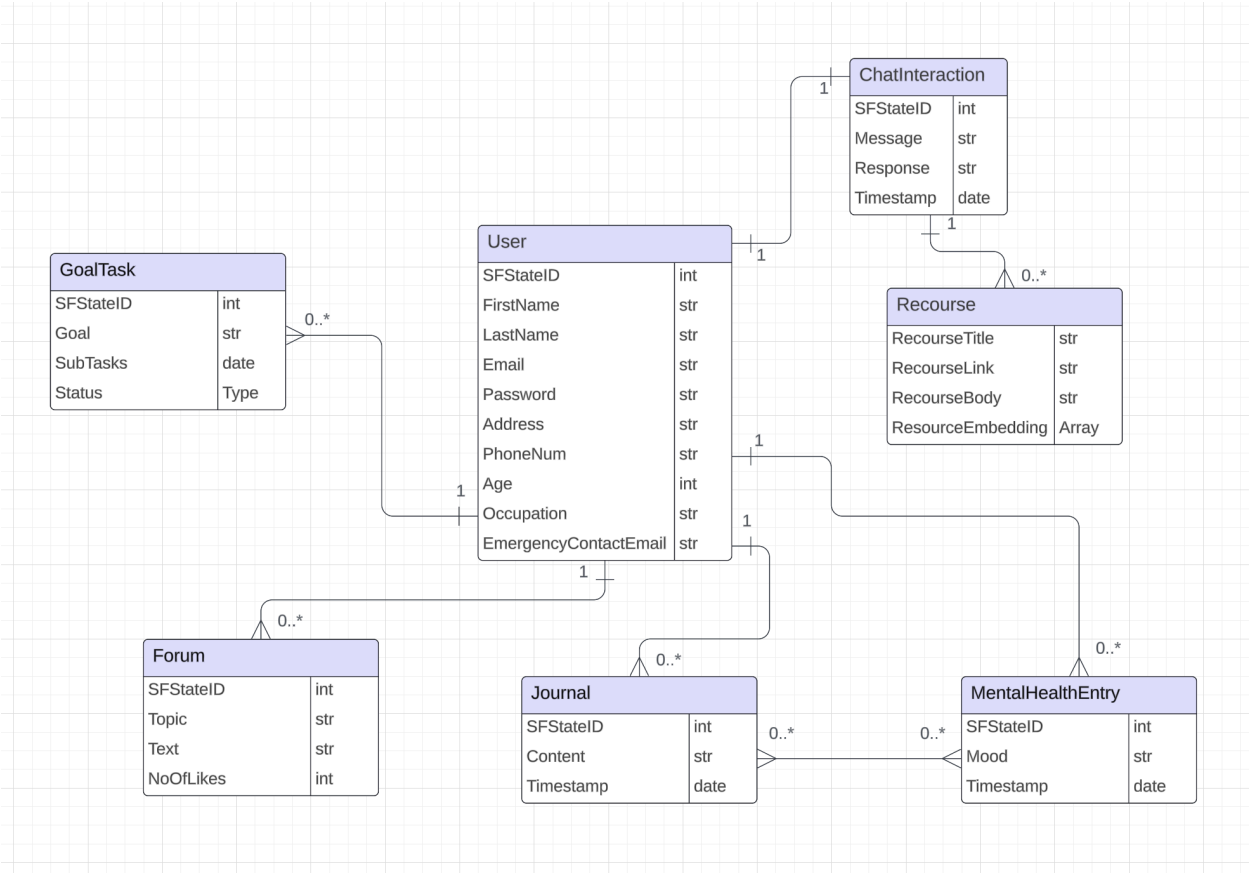
6. **Journal**: {
   "SFStateID": {
    "$type": "int",
    "description": "9-digit number (same as the User's SFStateID)"
   },
   "Content": {
    "$type": "string",
    "description": "The journal entry content provided by the user"
   },

```
    "Timestamp": {
      "$type": "date",
      "description": "Timestamp when the journal entry was created"
    }
  }
```

7. **Forum**: {
```
    "SFStateID": {
      "$type": "int",
      "description": "9-digit number (same as the User's SFStateID)"
    },
    "Topic": {
      "$type": "string",
      "description": "The topic of the forum post, e.g., advice, tips, question"
    },
    "Text": {
      "$type": "string",
      "description": "The content of the forum post"
    },
    "NoOfLikes": {
      "$type": "int",
      "description": "The number of likes the post has received"
    }
  }
```

Add/Delete/Search Architecture	Functional Requirement
Add/ Delete for Users	Users can add/ delete information in the profile section
Add/Delete for GoalTask	Users can add new goals or delete finished goals.
Add/Delete for Journal	Users can add new Content or delete existing content from the journal
Add/ Delete for Forum	Users can add new post or delete old post from Forum

5. High Level UML Diagrams



6. Identify actual key risks for your project at this time

- **Skills risks and mitigation plan**
    - Lacking experience with Figma for UI/UX Flow
        - Resolved by:
            - Practice creating mockups for current websites
            - Look up tutorials for Figma
    - Database Design unclear
        - Resolved by:
            - Go over DB organization thoroughly
            - Go over UML Diagram with backend members
    - Lacking skill in React, MongoDB, HTML,CSS, Python,etc.
        - Resolved by:
            - Practicing with other skilled members
            - Looking up tutorials
            - Asking ChatGPT for help

- **Schedule risks**
    - Conflicts with other classes for meetings/progress
        - Resolved by:
            - Communicating with members
            - Figuring out alternate meet times
            - Asking for summary to catch up incase missed meeting
    - UI/UX not complete, delaying progress for frontend or backend
        - Resolved by:
            - Frontend asking for help if needed
            - Regular updates on progress
            - Set tasks
    - Database not complete/unclear, delaying Virtual Prototype
        - Resolved by:
            - Discussing with frontend if needed
            - Improve clarity on data types
            - Regular check ins with teams for explanations and feedback
- **Teamwork risks**
    - Overall design is vague or unclear can cause alignments and unnecessary work
        - Resolved by:
            - Improve clarity on design with whole team
            - Regular team meetings to go over design
            - Remove unnecessary things
    - Lack of communication between front and back end causes overlapping work due to misalignment
        - Resolved by:
            - Can host meetings besides the whole team meetings
            - Make sure everyone is caught up with APIs
            - Keep consistent updates between each other
            - Make sure we are using most up to date version of website
- **Legal/content risks**
    - Using copyrighted content, leads to legal and reputation issues
        - Resolved by:
            - Double checking images, content, etc if copyrighted
            - Making sure to give credit to source if allowed
            - Try to use copyright free content
            - Inform team about copyright laws

7. Project Management

We have each of our member's progress shared in the Discord server, as well as verbally during scrum meetings. All tasks are written and sent in Discord with specific *#frontend-to-do* and *#backend-to-do* channels for scope specific tasks, as well as a general *#to-do-list* for major tasks. When we are done with the lists, we send a Done message.