SW Engineering CSC648-01 Summer 2024

CodeConnect

Team 1 - PikaDevs
Max Shigeyoshi - mshigeyoshi@sfsu.edu - Team Lead
Aaron Rayray - arayray@sfsu.edu - Github Master
Noah Hai - nhai@sfsu.edu - Doc Editor
Shez Rahman - srahman2@sfsu.edu - Backend Lead
Ghadeer Al-Badani - galbadani@sfsu.edu - Backend
Majd Alnajjar - malshemari@sfsu.edu - Frontend
William Pan - wpan1@sfsu.edu - Database Admin
Phillip Ma - pma1@mail.sfsu.edu - Frontend Lead

"Milestone 3"

6/26/2024

History Table:

Date	Changes

1. Data Definitions

1. User : Class - The user that creates the account.
   1.1. userID : Number - Primary key
   1.2. firstName : String
   1.3. lastName : String
   1.4. userName : String
   1.5. membershipType : String-Symbol (FREE, PREMIUM, MENTOR)
   1.6. email : String
   1.7. password : Number
   1.8. salt : Number - random data for hashing
   1.9. emailVerified : Boolean
   1.10. resetPasswordToken: String
   1.11. resetPasswordExpires : Date
   1.12. points : Number
   1.13. rankID : Number - Rank object
   1.14. challengesCompleted : Number - Array of Numbers(completed challenge IDs)
   1.15. numChallengesCompleted : Number
   1.16. allTrophies : JSON- Array of SpecificTrophy objects
   1.17. streakChallenge: Number - Streak of challenges completed
   1.18. coins : Number
   1.19. mentees : Json - Array of userIDs
   1.20. notificationList :Json - Array of Notification objects
   1.21. bookmarks : String - Array of postIDs
   1.22. numPosts : Number - number of posts + number of comments
   1.23. groups : Json - Array of Group objects
   1.24. groupsMentored : Json - Array of MentorGroup objects
   1.25. isPremium : boolean
   1.26. isMentor : boolean

2. PremiumUser : Class, extends User : A user that has paid the subscription fee for premium features
   2.1. userID : Number - Primary and foreign key

3. MentorUser : Class, extends User - A user that has become a mentor to other users. Must be approved by interview.
   3.1. userID : Number - Primary and foreign key
   3.2. feedbackCompleted: String- Array of FeedbackForm objects

4. Profile : The user’s profile which has information about the user.
    4.1. profileID: Number - Primary key
    4.2. userID: Number - foreign key (the user who owns the profile)
    4.3. isMentor : boolean
    4.4. biography : String
    4.5. resume : String
    4.6. portfolioID : Number

5. ExternalLinks : Class - contains external links to be used by profile
    5.1. externalLinksID: Number - Primary key
    5.2. profileID : Number - foreign key
    5.3. xLogo: String - filepath to image
    5.4. linkedinLogo: String - filepath to image
    5.5. instagramLogo: String - filepath to image
    5.6. tiktokLogo: String - filepath to image
    5.7. facebookLogo: String - filepath to image
    5.8. xLink: String
    5.9. linkedinLink: String
    5.10. instagramLink: String
    5.11. tiktokLink: String
    5.12. facebookLink: String
    5.13. hasX: Boolean
    5.14. hasLinkedin: Boolean
    5.15. hasInstagram: Boolean
    5.16. hasTiktok: Boolean
    5.17. hasFacebook: Boolean

6. Portfolio: Class - Users will have this to display their projects
    6.1. portfolioID : Number - Primary key
    6.2. userID : Number - foreign key (user that create portfolio)
    6.3. visibility: Symbol (public or private)

7. Project : Class - these are data items to be stored in portfolio
    7.1. projectID : Number - Primary key
    7.2. Portfolio : Number - Foreign key
    7.3. link: String - link to project
    7.4. desc: String - text description of project
    7.5. title: String - title of project
    7.6. pictures: String - Array of file paths to images

8. Notification: Class
    8.1. notificationID : Number - primary key
    8.2. title: String - title of notification with template
    8.3. redirectLink: String - clicking notification takes you to link
    8.4. date: Date
    8.5. time: Time

9. UserNotification: A junction table(Associative entity) for notification and user
   Since a user has many notification and a notification belong to many user
    9.1. userNotificationID: Number - primary key
    9.2. userID: Number - foreign key
    9.3. notificationID: Number - foreign key

10. Group : Class - users can join these to bond over commonalities such as being alumni from the same school, or having interest in a certain technology
    10.1. GroupsID: Number - primary key
    10.2. allMembers: String - Array of User objects who have access
    10.3. forum: String - Forum object
    10.4. groupsID : Number - foreign key

11. UserGroup: A junction table(Associative entity) for group and user
    Since a mentorUser can join many MentorGroup and A MentorGroup can have many mentorUser
    11.1. UserGroupID: Number - primary key
    11.2. groupID: Number - foreign key reference group
    11.3. userID: Number - foreign key reference User

12. MentorGroup : Class extends Group - a group for mentors to communicate with their mentees
    12.1. groupID: Number - primary key
    12.2. mentorMembers: String - Array of User objects
    12.3. groupsID : Integer - foreign key

13. UserMentorGroup: A junction table(Associative entity) for group and user
    Since a mentorUser can join many MentorGroup and A MentorGroup can have many mentorUser
    13.1. UserMentorGroupID: Number - primary key
    13.2. groupID: Number - foreign key reference Mentorgroup
    13.3. userID: Number - foreign key reference mentorUser

14. Groups : Class - a list of all groups for access
    14.1. GrouopsID: Number - primary key
    14.2. mentorGroups : String - Array of MentorGroup objects
    14.3. groups:String - Array of Group objects

15. Post : Class - Posts that users can create in order to interact with the larger community.
    15.1. postID: Number - primary key
    15.2. userID: Number - foreign key
    15.3. content : String - the text content
    15.4. comments: String - Array of comments/replies to this post
    15.5. codeBlock: String
    15.6. date: Date
    15.7. time: Time
    15.8. likes: Number - number of likes on the post
    15.9. threadID : foreign key

16. Forum : Class - A collection of posts and data about the forum
    16.1. forumID: Number - primary key
    16.2. threadID : Number - foreign key
    16.3. threadTitle: String - Title of the forum thread
    16.4. date: Date - object
    16.5. time: Time - object
    16.6. access : String - List of members who have access
    16.7. threads: String - Array of ForumThread objects

17. ForumThread: Class - used to contain all posts under a thread topic
    17.1. threadID: Number - label each thread in unique identifier
    17.2. originalPoster: String - User object who started the thread
    17.3. threadTitle: String - title of the forum thread
    17.4. posts: Array of Post objects which make up the thread
    17.5. date: Date - object
    17.6. time: Time - object
    17.7. forumID: Number - foreign key

18. CodeChallenge : Class - The coding challenges that each user has access to and can attempt to solve.
    18.1. challengeID : Number - primary key
    18.2. title: String
    18.3. description: String

18.4. language: String
18.5. difficulty: String
18.6. codingBlock: String
18.7. deadline: Date - object
18.8. completionPoints: Number - points gained for completing this challenge
18.9. solutions: String - array of ChallengeSubmission objects - record of X successful solutions
18.10. codingTests: String
18.11. pseudocodeHint: String

19. UserChallenge: A junction table(Associative entity) for user and codeChallenge. Since a user has many codeChallenge and A codeChallenge belong to many user
    19.1. userChallengeID : Number - primary key
    19.2. userID: Number - foreign key
    19.3. challengeID: Number - foreign key

20. ChallengeSubmission: Class - contains submission data
    20.1. challengeSubID:Number -primary key
    20.2. userID: Number - foreign key
    20.3. challengeID: Number - foreign key
    20.4. codSub: string- the code is submitted
    20.5. date: Date - object
    20.6. time: Time - object
    20.7. verifiedSolution: Boolean - true if the submission was successful

21. SubmissionShare : Class (FR 1.18) - unique class to share with peers when you complete a challenge
    21.1. shareID : Number - primary key
    21.2. userID : Number - foreign key
    21.3. likes: Number
    21.4. Comments : String

22. Feedback : Class - The review form that mentors use to give feedback to mentees on their coding challenge solutions.
    22.1. feedbackID : Number - primary key
    22.2. userID : Number - foreign key reference mentorUser
    22.3. challengeSubID : Number - foreign key reference challengeSubmission
    22.4. solutionSubmission: String - ChallengeSubmission object the feedback is in response to

22.5. organizationFeedback: String - mentor’s written feedback on formatting/organization of code
22.6. organizationScore: Number - mentor’s scoring on scale of 1-5
22.7. logicFeedback: String - mentor’s written feedback on code logic and efficiency
22.8. logicScore: Number - mentor’s scoring on scale of 1-5
22.9. commentFeedback: String - mentor’s written feedback on code comments
22.10. commentScore: String - mentor’s scoring on scale of 1-5

23. Leaderboard : Class - list of all users organized by ranking points to be displayed as a table
23.1. leaderboardID: Number - primary key
23.2. userID: Number - foreign key
23.3. numPoints : Number - Each user’s number of points held
23.4. rankTitle: String - Each user’s rank title
23.5. rankIcon: String - corresponding icon for user ranking

24. Rank : Class - the different icons and titles that users can acquire as they gain more points
24.1. icon : String - path to image file
24.2. title : Symbol
24.3. rankID: Number - primary key
24.4. ranksID: Number -foreign key
24.5. pointsRange: Number - function returning true if User collected points fall in the range

25. Ranks : Class - list of different rank objects that users can acquire as they gain more points, and functions involving the ranks
25.1. ranksID : Number - primary key
25.2. rankID : Number - foreign key
25.3. checkRequirements: String - function to check what rank User has and return the correct rank

26. Trophy : Class - inherited by SpecificTrophy class. Users earn these for specific achievements
26.1. name: String
26.2. trophyID: Number - primary key
26.3. description: String - describes requirements to earn trophy
26.4. Trophy flag : Boolean - true if user possesses trophy
26.5. userID: Number - foreign key

26.6. checkRequirements : String- abstract method, implemented by SpecificTrophy

27. SpecificTrophy: Class extends Trophy - users can earn trophies for different achievements
    27.1. trophyID: Number - primary key
    27.2. hasTrophy : Boolean - used by User, is 1 if checkRequirements returns checkRequirements : function to be implemented

28. JobList : Class - Job listings that are available for any user to apply for.
    28.1. userID: Number - foreign key
    28.2. jobID: Number - primary key
    28.3. title: String
    28.4. company: String
    28.5. location: String
    28.6. description: String
    28.7. requirements: String
    28.8. date: Date - object
    28.9. applicationLink: String

29. userView: Class - A junction table(Associative entity) for joblist and user Since a user can view many jobList and a jobList belong to many user
    29.1. userViewID:Number - primary key
    29.2. userID: Number -foreign key
    29.3. jobId: Number - foreign key

30. PaymentInfo: Class - Users will be able to store their payment information for purchases
    30.1. paymentID: Number- primary key
    30.2. cardNumber: Number
    30.3. cardName: String
    30.4. zipCode: Number
    30.5. backCode: Number - CVV/CVC code
31. UserPayment: A junction table(Associative entity) for payment and user Since a user has many paymentInfo and a paymentInfo belong to many user
    31.1. userPaymentID: Number - primary key
    31.2. userID: Number - foreign key
    31.3. paymentID: Number - foreign key

32. Message : Class - Sent between users as direct messaging
    32.1. MessageID: Number -primary key
    32.2. userID: Number - who’s sending the message (foreign key)
    32.3. inboxID: Number - who’s receiving the message(foreign key)
    32.4. time: Time- object
    32.5. date: Date - object
    32.6. content: String - content of the message
    32.7. messageID: Number -foreign key

33. MessageThread: Class - includes all past replies to a single message thread
    33.1. messageThreadID : Number - primary key
    33.2. participatingUsers: String - Array of User objects who are in the message thread

34. Inbox : Class - Every user contains an inbox of messages that other users have sent them
    34.1. inboxID: Number - primary key and foreign key
    34.2. messageThreads:String - array of MessageThread objects

35. SupportForm : Class - All users can use these to communicate with the company
    35.1. supportFormID : Number - primary key
    35.2. from_userID : Number - foreign key reference user
    35.3. to_userID : Number - foreign key reference userHiring
    35.4. date: Date - object
    35.5. time: Date - object
    35.6. message: String - populated by user from UI

36. userHiring: Class extends User - A company user that hiring user
    36.1. userID : Number : primary key
    36.2. company : String - the company that user work for
    36.3. [1]position : String - The position of user is in company
37. chatSession : Meeting rooms which users can use to meet with other users (free or premium). (Assuming this relates to workspace idea - what separates this from an inbox thread with multiple recipients?)
    37.1. chatSessionID: Number - primary key
    37.2. User ID : Number - foreign key
    37.3. date : Date - object
    37.4. title : String

1

37.5. invitees : Json - list of user_ID’s

38. UserChatSession: A junction table(Associative entity) for user and chatSession Since a user has many codeChallenge and A codeChallenge belong to many user
    38.1. userChatSessionID: Number - primary key
    38.2. userID: Number - foreign key
    38.3. chatSessionID: Number - foreign key

39. Chatbot : alternative to support form for users to communicate with an AI rep for the company
    39.1. Chatbot ID: Number - primary key
    39.2. User ID : Number - foreign key reference userHiring

2. Functional Requirements

Priority 1:

1. **All Users:**

*1.1 Users shall be able to explore some portions of the product without a profile
*1.2 Users shall be able to solve an example problem.
*1.3 Users shall create a profile
*1.4 Users shall be able to Log in/Log out (only with created profile)
*1.6 Users shall be able to upload profile picture
*1.7 Users shall be able to Delete profile
*1.8 User shall be able to update payment information
*1.9 Users shall be able to make their profile private or public
*1.12 Users shall be able to check other users profiles/stats
*1.13 Users shall be able to do coding challenges
*1.15 Users shall be able to award different profile trophies for achievements
*1.17 Users shall be able to earn points for coding challenges
*1.18 Users shall be able to like/comment on challenge posts
*1.20 Users shall be able to check their coding ranking
*1.21 Users shall be able to check leaderboards
*1.22 Users shall be able to subscribe to Premium
*1.23 Users shall be able to unsubscribe to Premium
*1.32 Users shall be able to display other socials on their profiles
*1.34 Users shall be able to request additional features from the dev team
*1.35 Users shall be able to utilize live chat w/ other users
*1.36 Users shall be able to direct message other users
*1.38 Users shall be able to check coding streak counter of daily challenges
*1.39 Users shall be able to create a portfolio
*1.40 Users shall be able to check/update portfolio

*1.41 Users shall be able to change portfolio visibility (public/private)
*1.42 Users shall be able to access portfolio review
*1.45 Users shall be able to submit support forms to submit feedback regarding the app
*1.49 Users shall be able to view featured users (spotlight user that's completed most challenges/stayed the most active for the past month. Featured user gets changed monthly)
*1.54 Users shall be able to join group session workshops led by mentors
*1.60 Users shall be able to get assessed to become a mentor
*1.61 Users shall be able to use an IDE without having to create an account
*1.65 Users without premium shall be able to view three solutions per month
*1.66 Users shall be able to see the difference between premium and free membership perks
*1.67 Users shall be able to gain points by starting forum threads
*1.68 Users shall be able to gain points by commenting in threads
*1.69 Users shall be able to gain points by gaining friends
*1.70 Users shall be able to gain points by gaining mentees
*1.71 Users shall be able to gain points by gaining mentors
*1.72 Users shall be able to post text and images to their profiles
*1.73 Users shall be able to view a general feed of other users' updates and activities
*1.76 Users shall be able to search for other users
*1.77 Users shall be able to search for groups
*1.78 Users shall be able to receive notifications for new coding challenges.
*1.78 Users shall be able to receive notifications for messages and comments.
*1.79 Users shall be able to customize notification preferences.
*1.81 Users shall be able to save forum posts for later reading.
*1.82 Users shall be able to report inappropriate content.
*1.83 Users shall be able to block or mute other users.
*1.84 Users shall be able to set privacy settings for profile visibility.
*1.86 Users shall be able to request a mentor recommendation.
*1.90 Users shall be able to subscribe to notifications for specific forums or groups
*1.93 Users shall be able to view a scrolling list of job listings from home page
*1.94 Users shall be able to view other users' activity from their homepages

2. **<u>Free Users</u>**
   *2.2 Free Users shall be able to check code challenge repo
   *2.3 Free users shall be able to view three pseudocode hints per month

3. **<u>Premium Users</u>**
   *3.0 Premium Users shall be able to check a single system chosen solution after completing a challenge
   *3.2 Premium Users shall be able to request mentorship (premium)

*3.3 Premium Users shall be able to Match mentors with similar coding language experience
*3.4 Premium users shall be able to view one pseudocode hint per challenge
*3.7 Premium Users shall be able to request code review from a mentor

4. **Mentor Users**
*4.1 Mentor Users shall be able to review code solutions of other users they are mentoring.
*4.4 Mentor Users shall be able to review Free/Premium users resumes/portfolios
*4.5 Mentor users shall be able to complete challenge feedback on coding challenges completed by mentees
*4.6 Mentor users shall be able to gain points by completing challenge feedback on mentee solutions
*4.7 Mentor Users shall be able to upload videos
*4.8 Mentor users shall have their number of mentees displayed on their profiles
*4.9 Mentor users shall have their number of solutions reviewed displayed on their profiles
*4.10 Mentor users shall have the groups they lead displayed on their profiles

Priority 2:

1. **All Users:**
**1.11 Users shall be able to follow other users/mentors
**1.16 Users shall be able to allow switching coding languages for challenges (cross compatibility) (different compilers - maybe 2 or 3 priority here)(would prefer just different sets of challenges based on language)
**1.24 Users shall be able to see pop-up ads for premium/paid utilities
**1.48 Users shall be able to take mock interviews (practice interviews with real-time communication and feedback from peers and other users)
**1.51 Users shall be able to utilize discounts/offers on apps premium features
**1.54 Users shall be able to join group session workshops led by mentors
**1.56 Users shall be able to view/update their own calendar(scheduled hackathons, virtual meetings..) (API or creating our own calendar?)
**1.59 Users shall be able to utilize a button to change the screen to dark mode.
**1.63 Users shall be able to choose their preferred speaking language(How does that look in implementation?)
1.74 **Users shall be able to earn coins
1.75 **Users shall be able to trade coins for premium subscriptions
-Or top leaderboard users get a premium reward
1.80 **Users shall be able to bookmark coding challenges.

**1.87 Users shall be able to participate in live coding sessions.
**1.91 Users shall be able to access analytics on their coding performance.
**1.92 Users shall be able to receive notifications for new coding challenges.

2. **<u>Free Users:</u>**
   **2.1 Free Users shall be able to check the most average solutions after completing a challenge.

3. **<u>Premium Users:</u>**
   **3.1 Premium Users shall be able to check a system chosen set of 2-3 solutions after completing a challenge

4. **<u>Mentor Users:</u>**
   **4.3 Mentor Users shall be able to create coding challenges.

<u>Priority 3:</u>

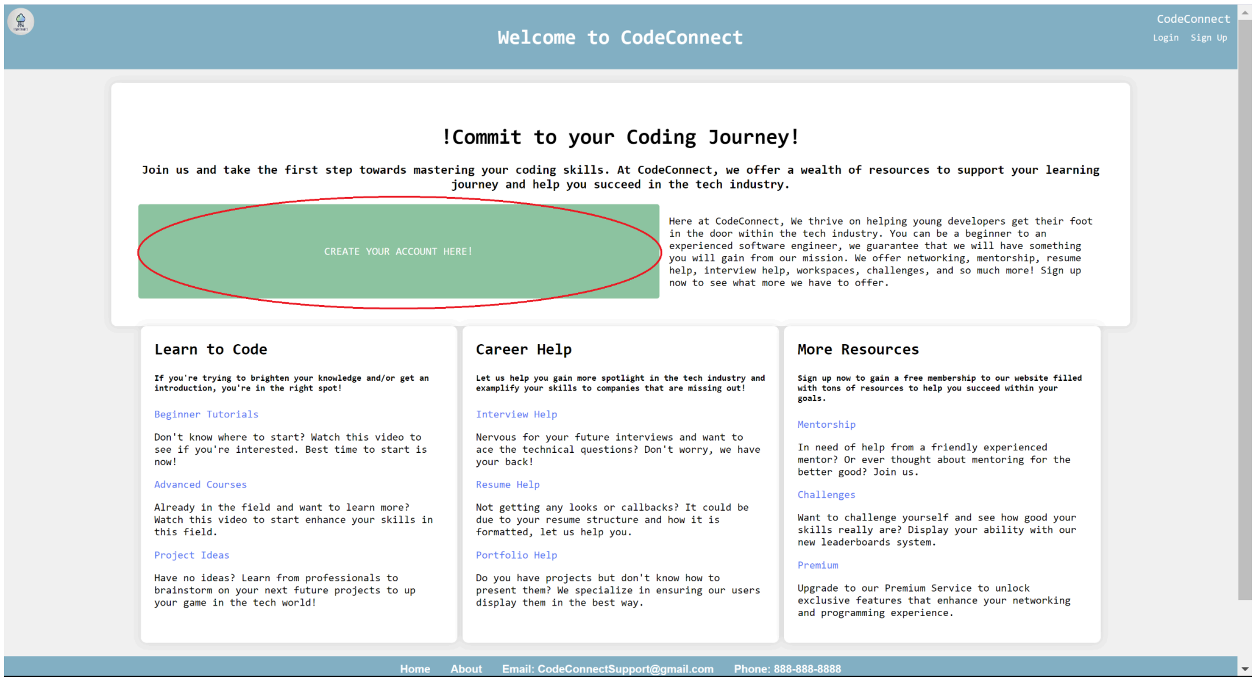
1. **<u>All Users:</u>**
   ***1.14 Users shall be able to see popular submissions (how is popular measured?)
   ***1.25 Users shall be able to application survey for mentorship (?)
   ***1.28 Users shall be able to take hiring questionnaire challenges (which are siblings to coding challenges
   ***1.37 Users shall be able to access free/paid resources (videos/textbooks)
   ***1.43 Users shall be able to utilize chatbot (this function would be cool but bold because need to know the data items) (Use support form instead)
   ***1.44 Users shall be able to access virtual workspace (define virtual workspace?)
   ***1.46 Users shall be able to open source/collaborative coding projects (users can participate in other people's coding projects) (Similar to 1.44?)
   ***1.47 Users shall be able to read weekly digests (recommendations created for users based on interests/what they worked on)
   ***1.50 Users shall be able to utilize pair programming sessions (allows users to work on projects/code collaboratively) (similar to 1.44 and 1.46)
   ***1.52 Users shall be able to read technical news (similar to 1.47)
   ***1.57 Users shall be able to create Recruiter/Hiring Manager specific profile
   ***1.58 Users shall be able to pass a Recruiter/Hiring manager verification/background check
   ***1.88 Users shall be able to join virtual coding bootcamps.
   ***1.89 Users shall be able to create and manage a personal blog.

2. **<u>Free Users: N/A</u>**

1. **<u>Premium Users:</u>**
   ***3.5 Premium Users shall be able to create code challenge repo, which are reviewed by the dev team prior to publication. (with a full repo it could be harder to check solutions vs a simple IDE plug in/compiler. Resources may be difficult on this topic…maybe talk to prof)
   ***3.6 Premium Users shall be able to check/update code challenge repo(^see note for 3.5)
2. **<u>Mentor Users:</u>**
   ***4.2 Mentor Users shall be able to set up group work stations. (what is a group work station? Also potentially lower priority)

3. Wireframes Based on your Mockups/Storyboards (detailed)

Toby:



CodeConnect
Login Sign Up

Welcome to CodeConnect

!Commit to your Coding Journey!

Join us and take the first step towards mastering your coding skills. At CodeConnect, we offer a wealth of resources to support your learning journey and help you succeed in the tech industry.

CREATE YOUR ACCOUNT HERE!

Here at CodeConnect, We thrive on helping young developers get their foot in the door within the tech industry. You can be a beginner to an experienced software engineer, we guarantee that we will have something you will gain from our mission. We offer networking, mentorship, resume help, interview help, workspaces, challenges, and so much more! Sign up now to see what more we have to offer.

Learn to Code

If you're trying to brighten your knowledge and/or get an introduction, you're in the right spot!

Beginner Tutorials

Don't know where to start? Watch this video to see if you're interested. Best time to start is now!

Advanced Courses

Already in the field and want to learn more? Watch this video to start enhance your skills in this field.

Project Ideas

Have no ideas? Learn from professionals to brainstorm on your next future projects to up your game in the tech world!

Career Help

Let us help you gain more spotlight in the tech industry and examplify your skills to companies that are missing out!

Interview Help

Nervous for your future interviews and want to ace the technical questions? Don't worry, we have your back!

Resume Help

Not getting any looks or callbacks? It could be due to your resume structure and how it is formatted, let us help you.

Portfolio Help

Do you have projects but don't know how to present them? We specialize in ensuring our users display them in the best way.

More Resources

Sign up now to gain a free membership to our website filled with tons of resources to help you succeed within your goals.

Mentorship

In need of help from a friendly experienced mentor? Or ever thought about mentoring for the better good? Join us.

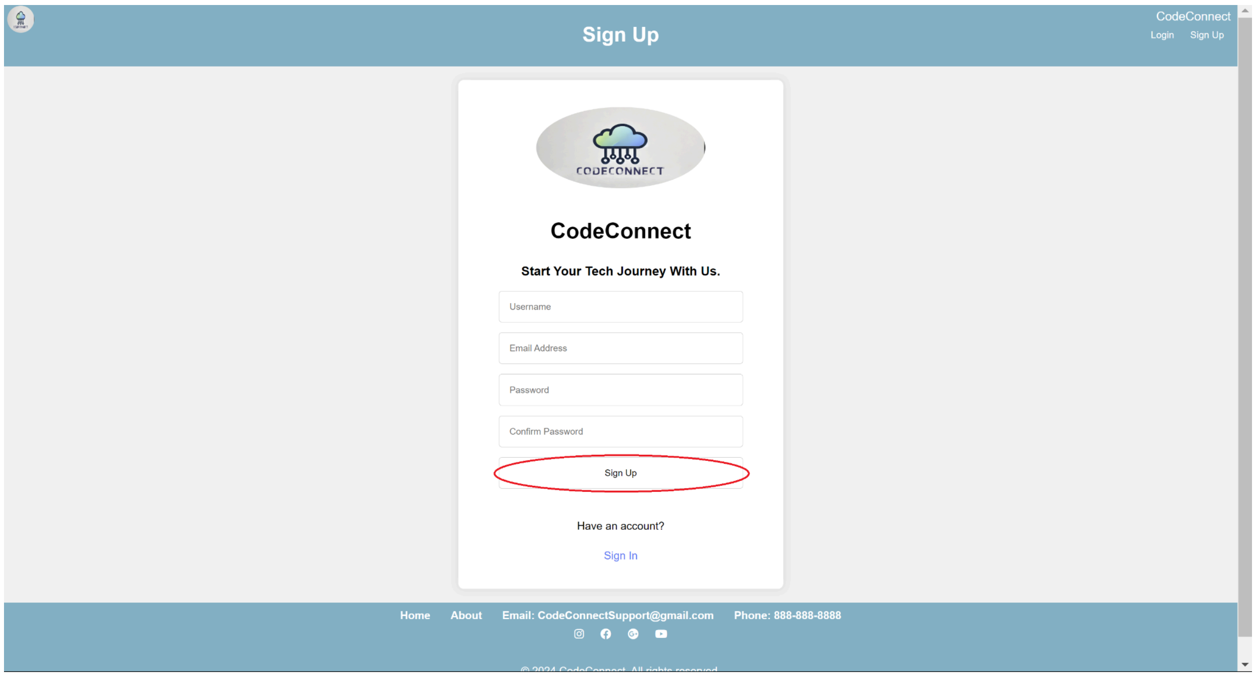
Challenges

Want to challenge yourself and see how good your skills really are? Display your ability with our new leaderboards system.

Premium

Upgrade to our Premium Service to unlock exclusive features that enhance your networking and programming experience.

Home About Email: CodeConnectSupport@gmail.com Phone: 888-888-8888



CodeConnect
Login Sign Up

Sign Up

CodeConnect

Start Your Tech Journey With Us.

Username

Email Address

Password

Confirm Password

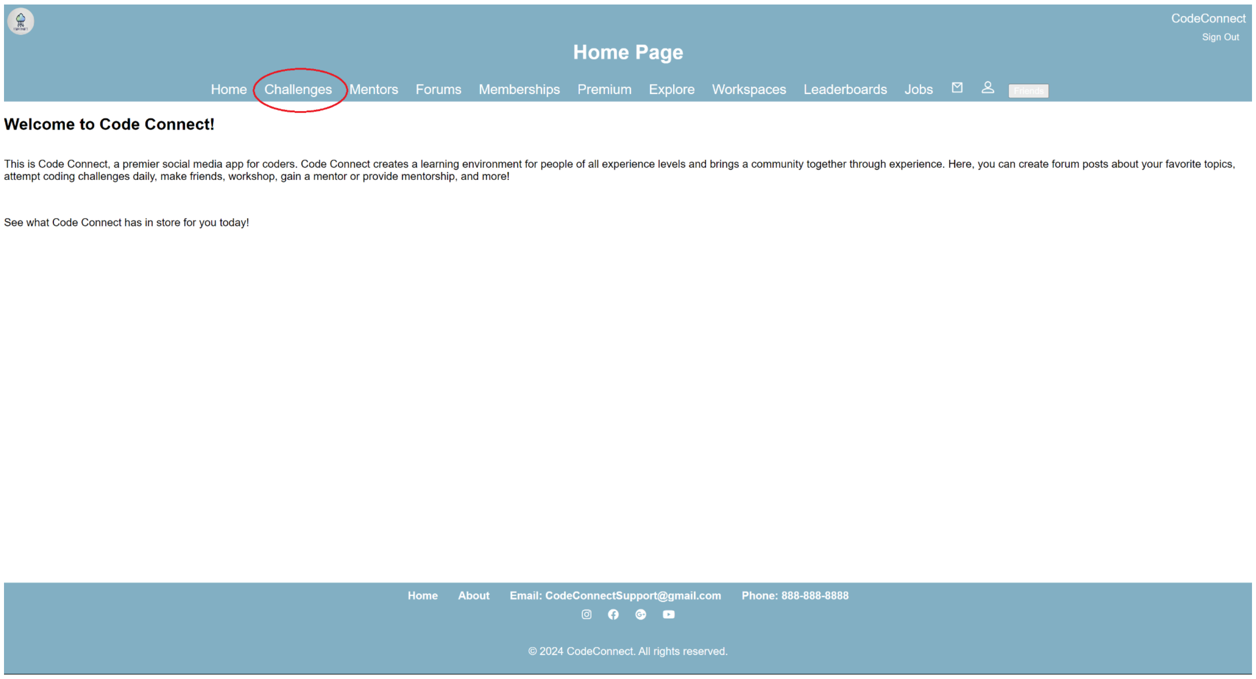
Sign Up

Have an account?

Sign In

Home About Email: CodeConnectSupport@gmail.com Phone: 888-888-8888

© 2024 CodeConnect. All rights reserved.


CodeConnect
Sign Out

Home Page

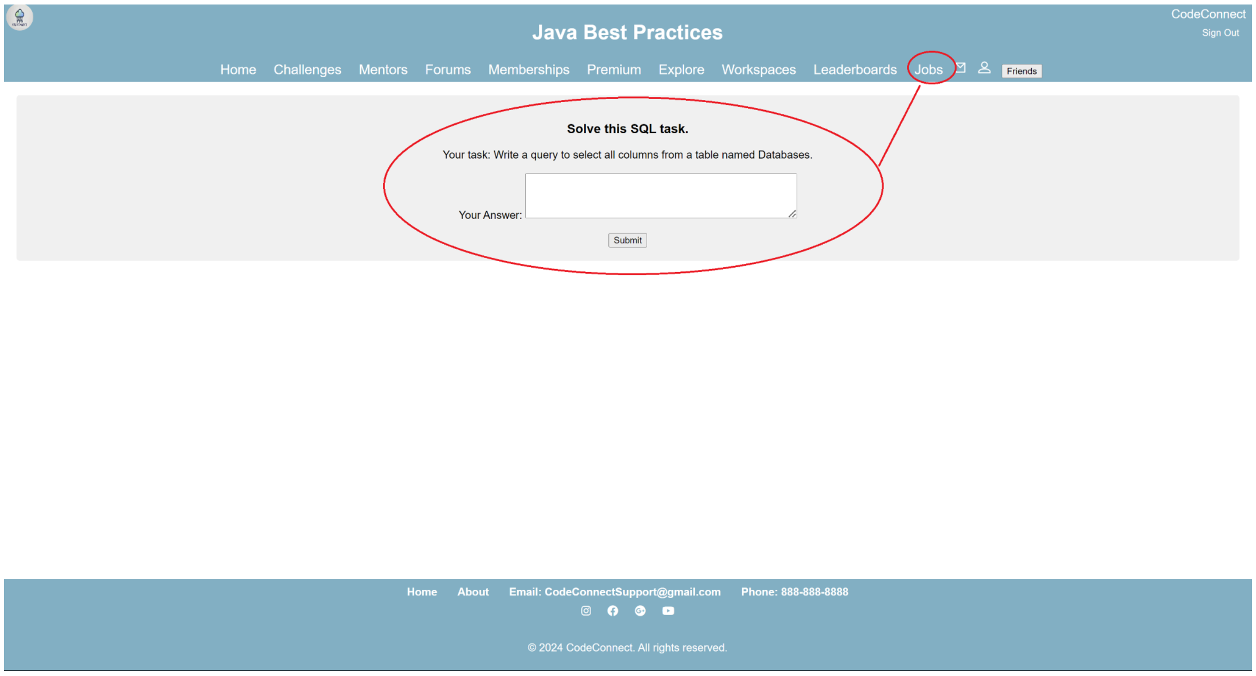
Welcome to Code Connect!

This is Code Connect, a premier social media app for coders. Code Connect creates a learning environment for people of all experience levels and brings a community together through experience. Here, you can create forum posts about your favorite topics, attempt coding challenges daily, make friends, workshop, gain a mentor or provide mentorship, and more!

See what Code Connect has in store for you today!

Home **About** **Email: CodeConnectSupport@gmail.com** **Phone: 888-888-8888**

© 2024 CodeConnect. All rights reserved.


CodeConnect
Sign Out

Java Best Practices

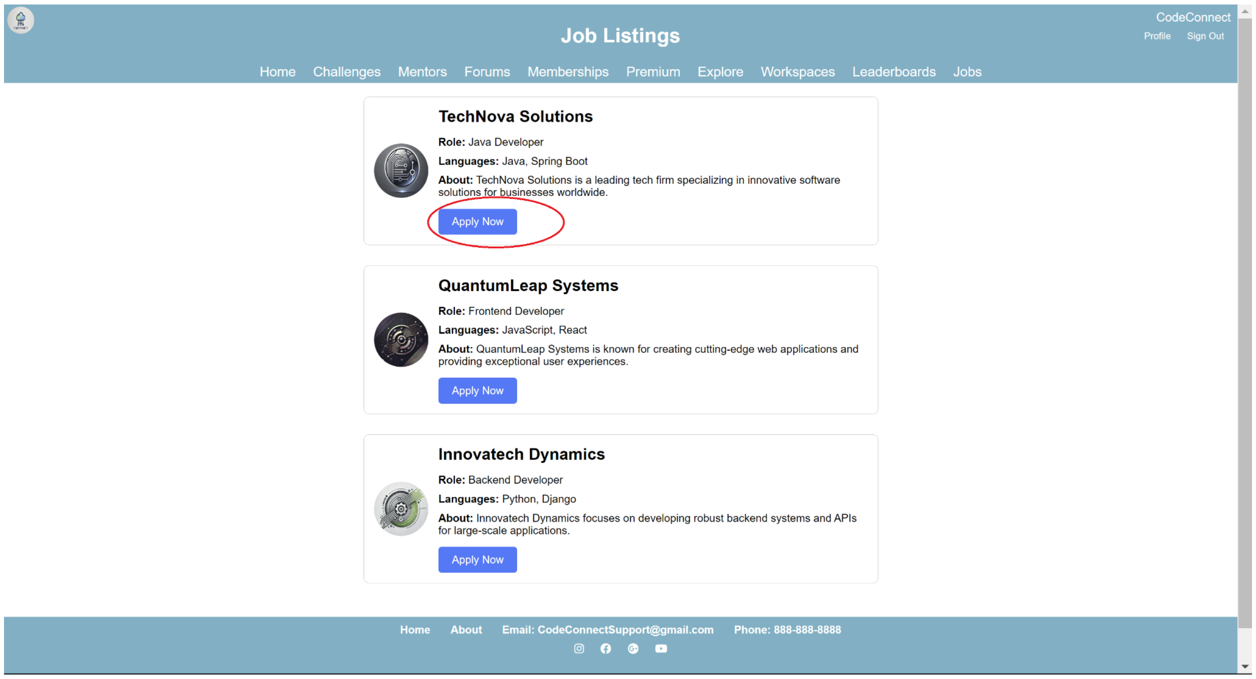
Solve this SQL task.

Your task: Write a query to select all columns from a table named Databases.

Your Answer:

Home **About** **Email: CodeConnectSupport@gmail.com** **Phone: 888-888-8888**

© 2024 CodeConnect. All rights reserved.



CodeConnect

Profile Sign Out

Job Listings

Home Challenges Mentors Forums Memberships Premium Explore Workspaces Leaderboards Jobs

TechNova Solutions

Role: Java Developer

Languages: Java, Spring Boot

About: TechNova Solutions is a leading tech firm specializing in innovative software solutions for businesses worldwide.

Apply Now

QuantumLeap Systems

Role: Frontend Developer

Languages: JavaScript, React

About: QuantumLeap Systems is known for creating cutting-edge web applications and providing exceptional user experiences.

Apply Now

Innovatech Dynamics

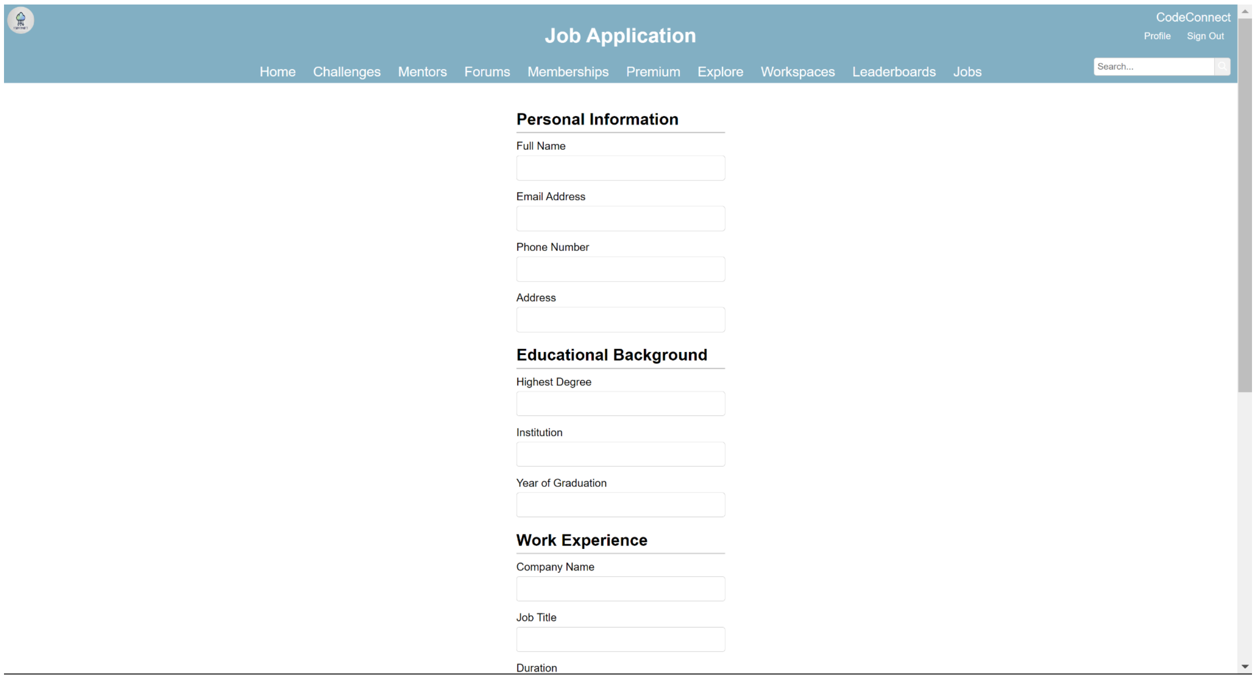
Role: Backend Developer

Languages: Python, Django

About: Innovatech Dynamics focuses on developing robust backend systems and APIs for large-scale applications.

Apply Now

Home About Email: CodeConnectSupport@gmail.com Phone: 888-888-8888



Job Application

Home Challenges Mentors Forums Memberships Premium Explore Workspaces Leaderboards Jobs

Personal Information

Full Name

Email Address

Phone Number

Address

Educational Background

Highest Degree

Institution

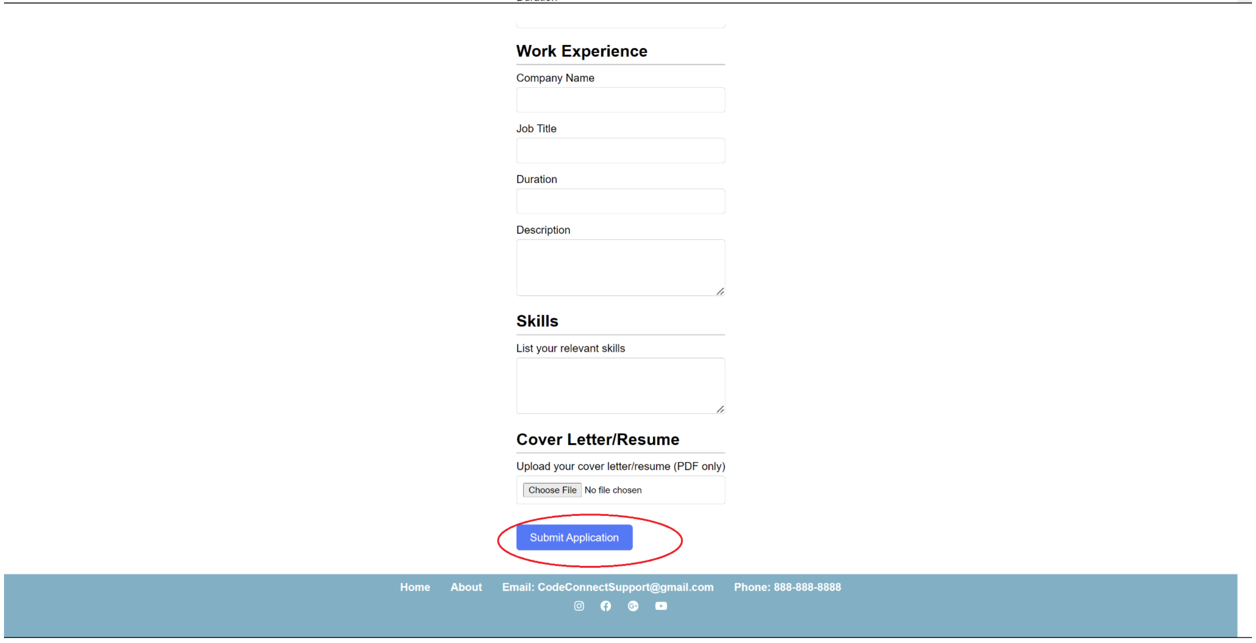
Year of Graduation

Work Experience

Company Name

Job Title

Duration



Work Experience

Company Name

Job Title

Duration

Description

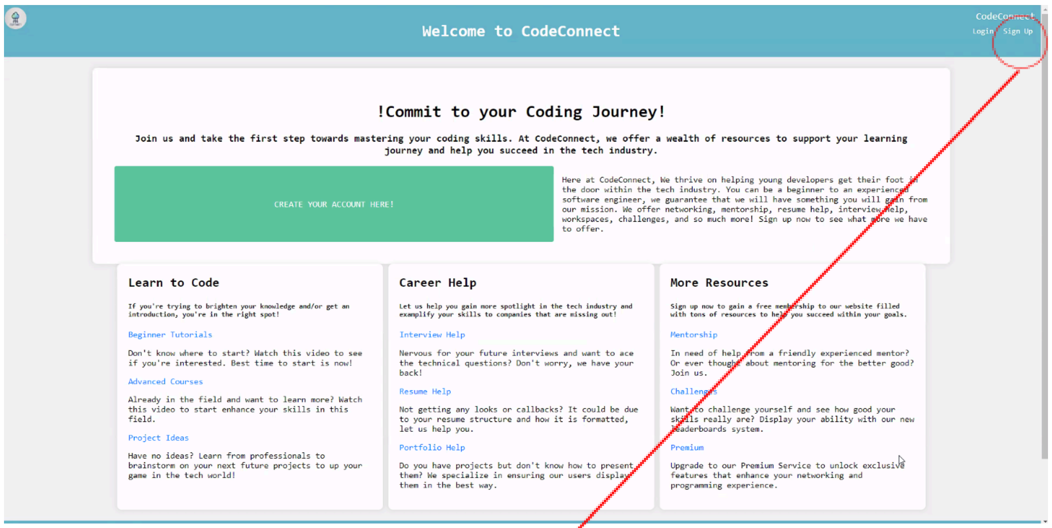
Skills

List your relevant skills

Cover Letter/Resume

Upload your cover letter/resume (PDF only)

Home About Email: CodeConnectSupport@gmail.com Phone: 888-888-8888


CodeConnect
Login Sign Up

Welcome to CodeConnect

!Commit to your Coding Journey!

Join us and take the first step towards mastering your coding skills. At CodeConnect, we offer a wealth of resources to support your learning journey and help you succeed in the tech industry.

CREATE YOUR ACCOUNT HERE!

Here at CodeConnect, We thrive on helping young developers get their foot in the door within the tech industry. You can be a beginner to an experienced software engineer, we guarantee that we will have something you will gain from our mission. We offer networking, mentorship, resume help, interview help, workspaces, challenges, and so much more! Sign up now to see what more we have to offer.

Learn to Code

If you're trying to brighten your knowledge and/or get an introduction, you're in the right spot!

Beginner Tutorials

Don't know where to start? Watch this video to see if you're interested. Best time to start is now!

Advanced Courses

Already in the field and want to learn more? Watch this video to start enhance your skills in this field.

Project Ideas

Have no ideas? Learn from professionals to brainstorm on your next future projects to up your game in the tech world!

Career Help

Let us help you gain more spotlight in the tech industry and examplify your skills to companies that are missing out!

Interview Help

Nervous for your future interviews and want to ace the technical questions? Don't worry, we have your back!

Resume Help

Not getting any looks or callbacks? It could be due to your resume structure and how it is formatted, let us help you.

Portfolio Help

Do you have projects but don't know how to present them? We specialize in ensuring our users display them in the best way.

More Resources

Sign up now to gain a free membership to our website filled with tons of resources to help you succeed within your goals.

Mentorship

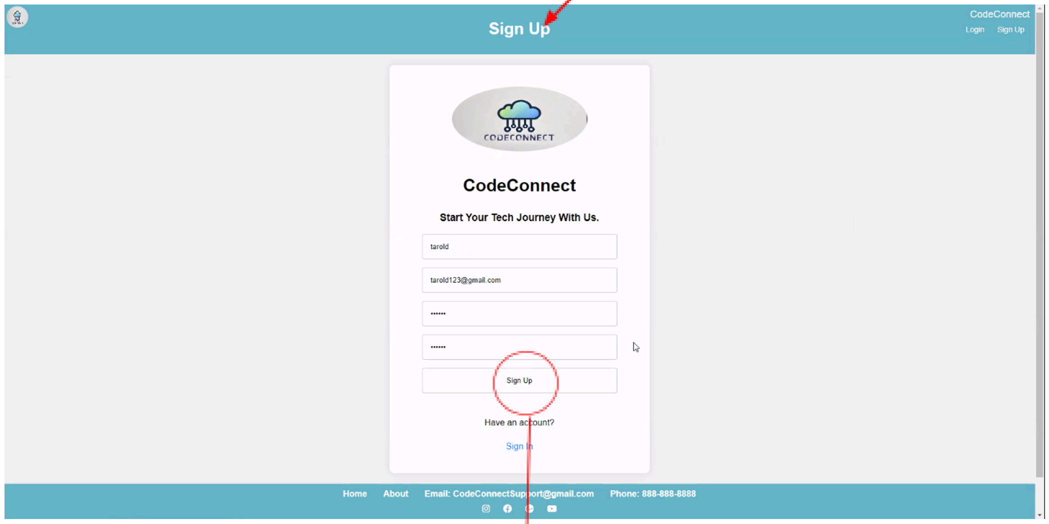
In need of help from a friendly experienced mentor? Or ever thought about mentoring for the better good? Join us.

Challenges

Want to challenge yourself and see how good your skills really are? Display your ability with our new leaderboards system.

Premium

Upgrade to our Premium Service to unlock exclusive features that enhance your networking and programming experience.


CodeConnect
Login Sign Up

Sign Up

CODECONNECT

CodeConnect

Start Your Tech Journey With Us.

tarold

tarold123@gmail.com

••••••

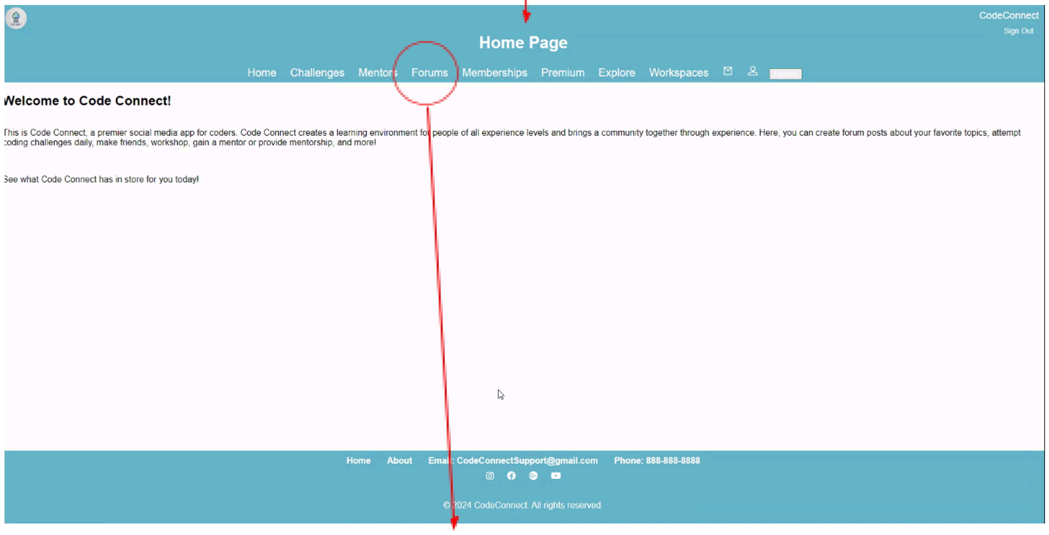
••••••

Sign Up

Have an account?

Sign In

Home About Email: CodeConnectSupport@gmail.com Phone: 888-888-8888


CodeConnect
Sign Out

Home Page

Home Challenges Mentors Forums Memberships Premium Explore Workspaces

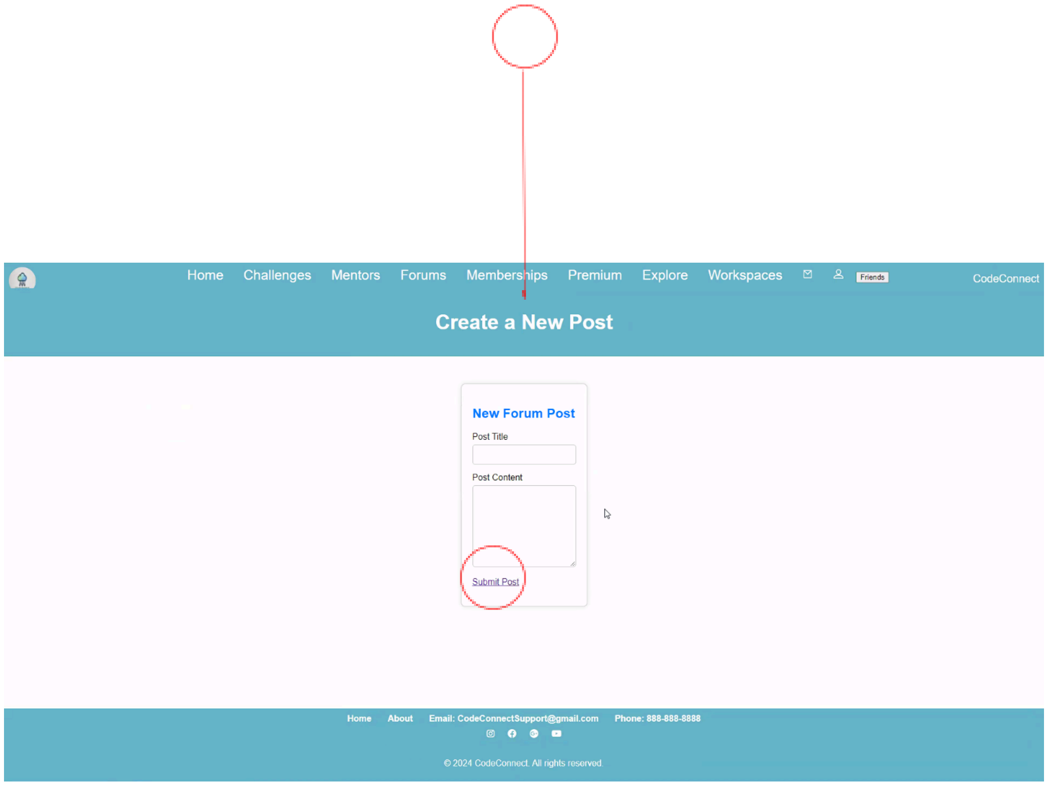
Welcome to Code Connect!

This is Code Connect, a premier social media app for coders. Code Connect creates a learning environment for people of all experience levels and brings a community together through experience. Here, you can create forum posts about your favorite topics, attempt coding challenges daily, make friends, workshop, gain a mentor or provide mentorship, and more!

See what Code Connect has in store for you today!

Home About Email: CodeConnectSupport@gmail.com Phone: 888-888-8888

© 2024 CodeConnect. All rights reserved.



Create a New Post

New Forum Post

Post Title

Post Content

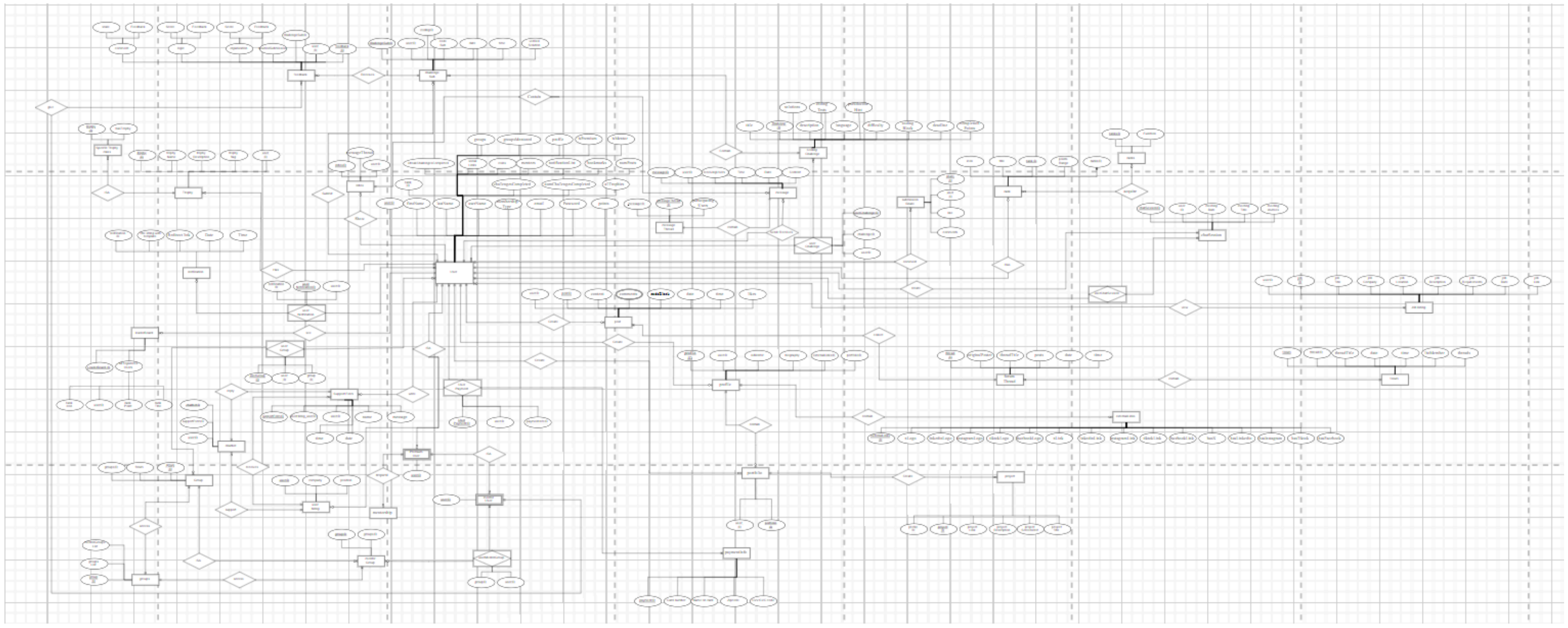
Email: CodeConnectSupport@gmail.com

Phone: 888-888-8888

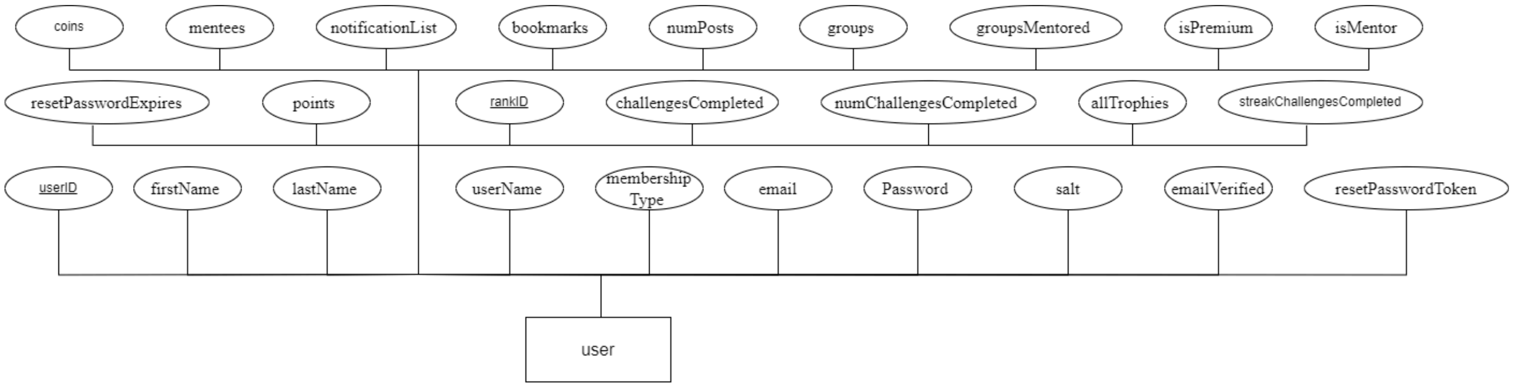
© 2024 CodeConnect. All rights reserved.

4. High level database architecture and organization (detailed) ERD:

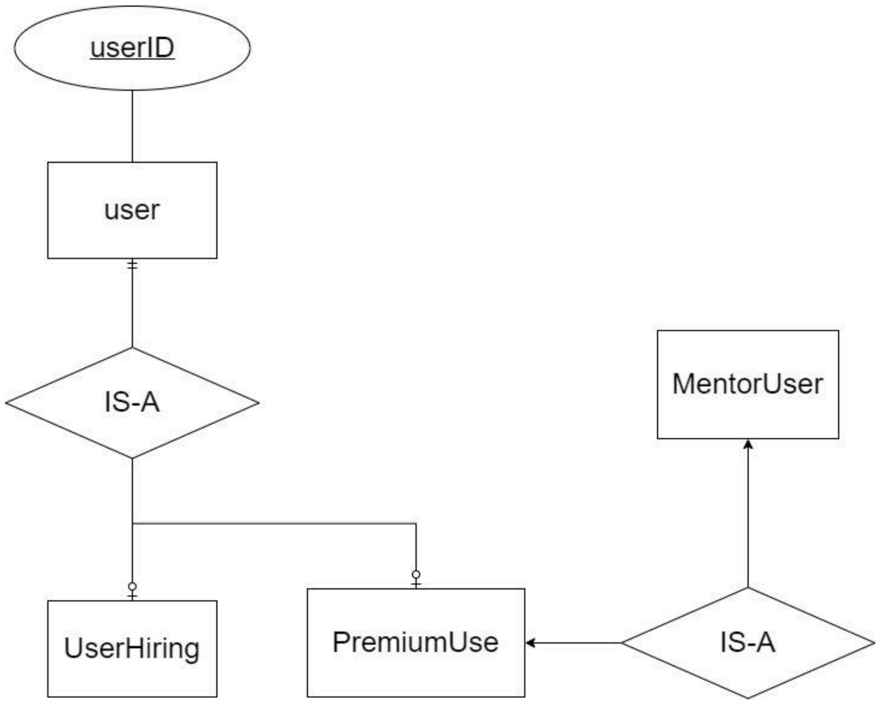
1) **overView** ERDV2 Draw Io link: https://drive.google.com/file/d/1zurpNjxRKnkSV0DKK_vZ_35LPtuBBGZS/view?usp=drive_link



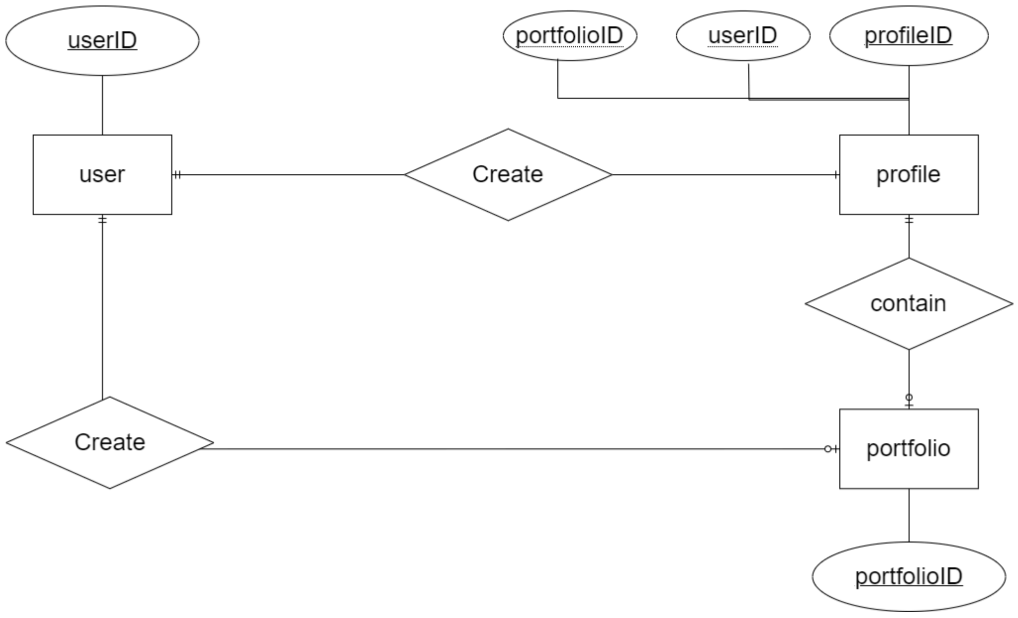
2) User table



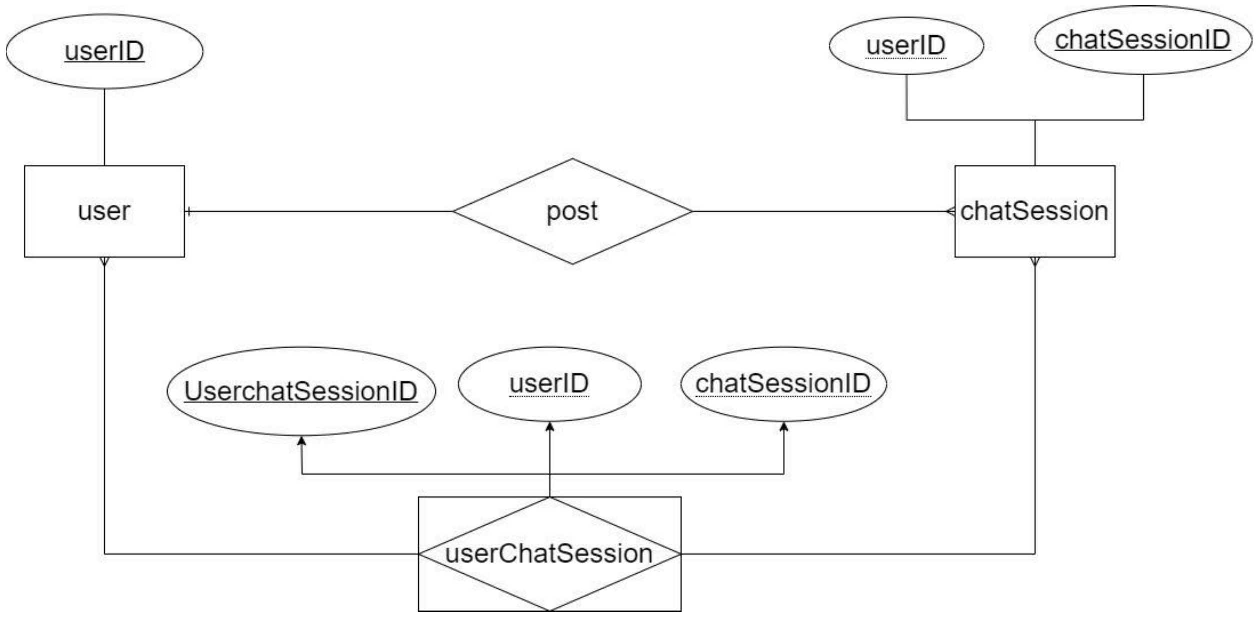
3) Inheritance



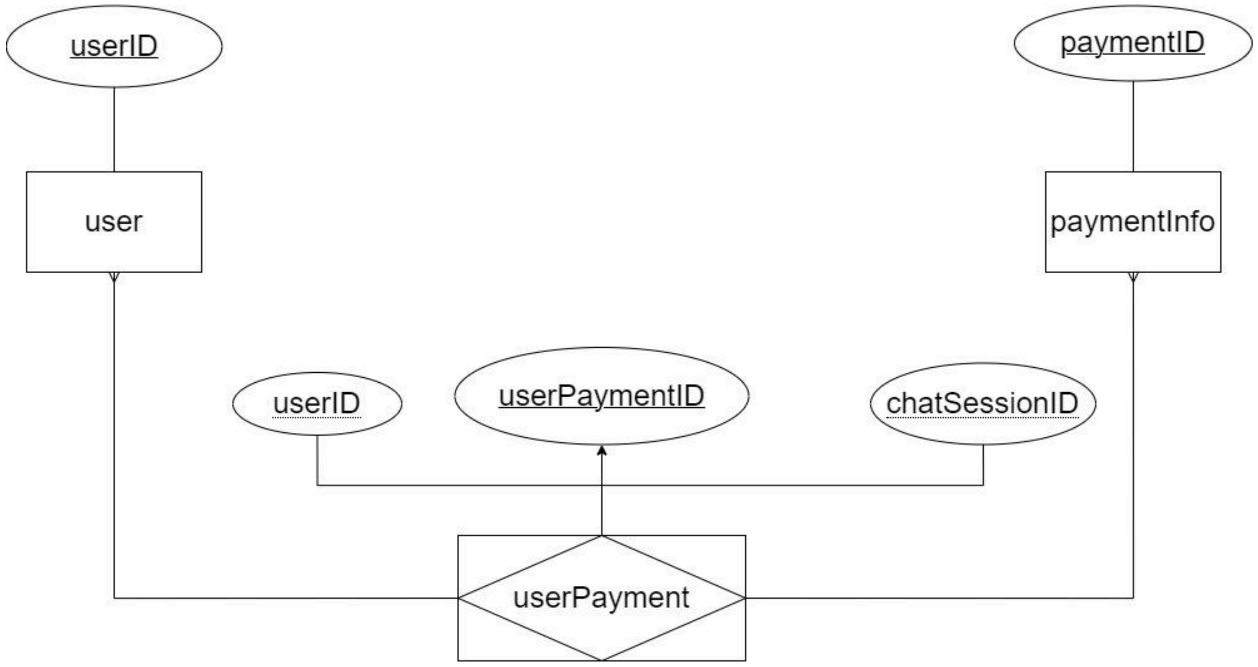
4) user, Profile and portfolio relationship



5) ChatSession and userChatSession

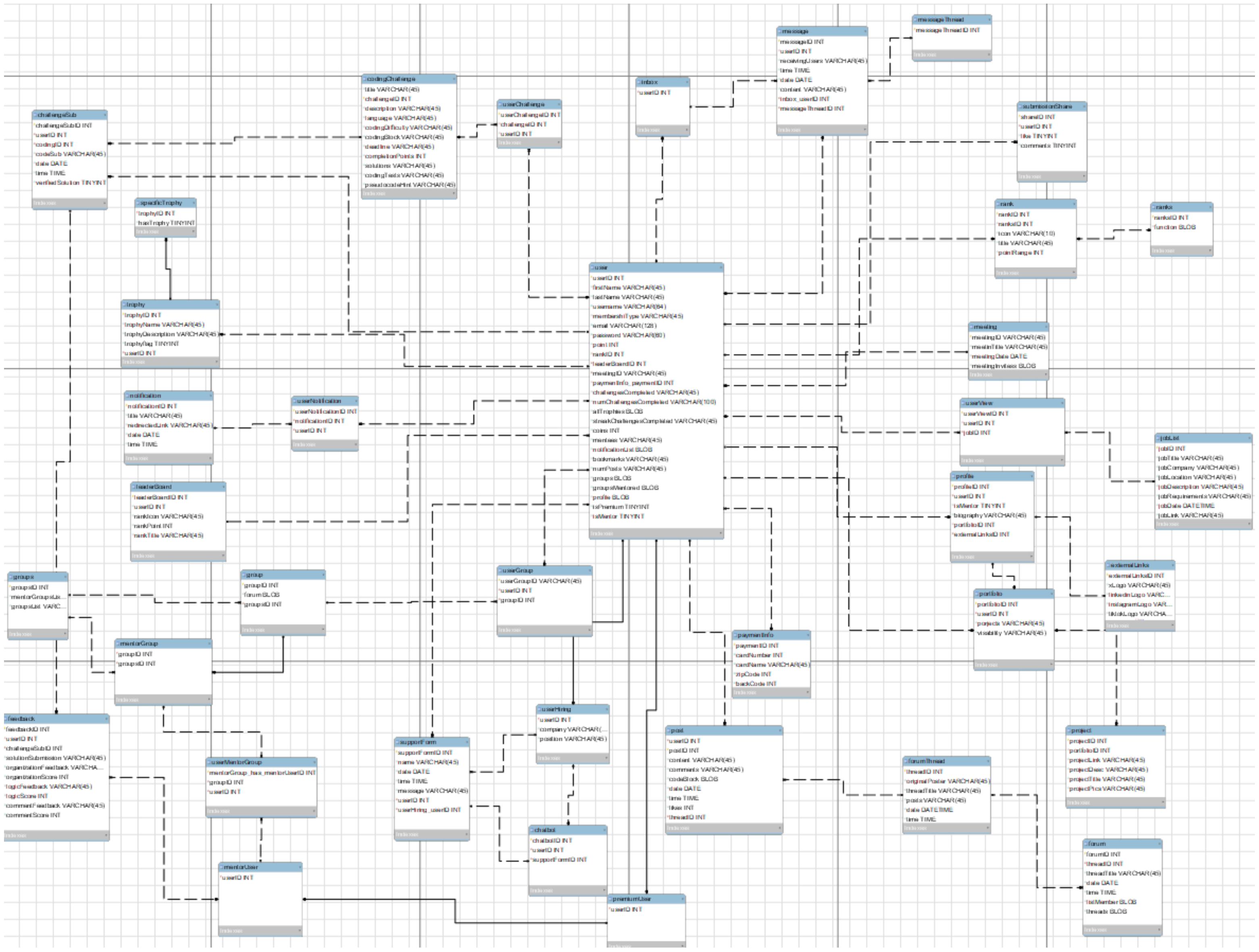


6) PaymentInfo and userPayment



EER:

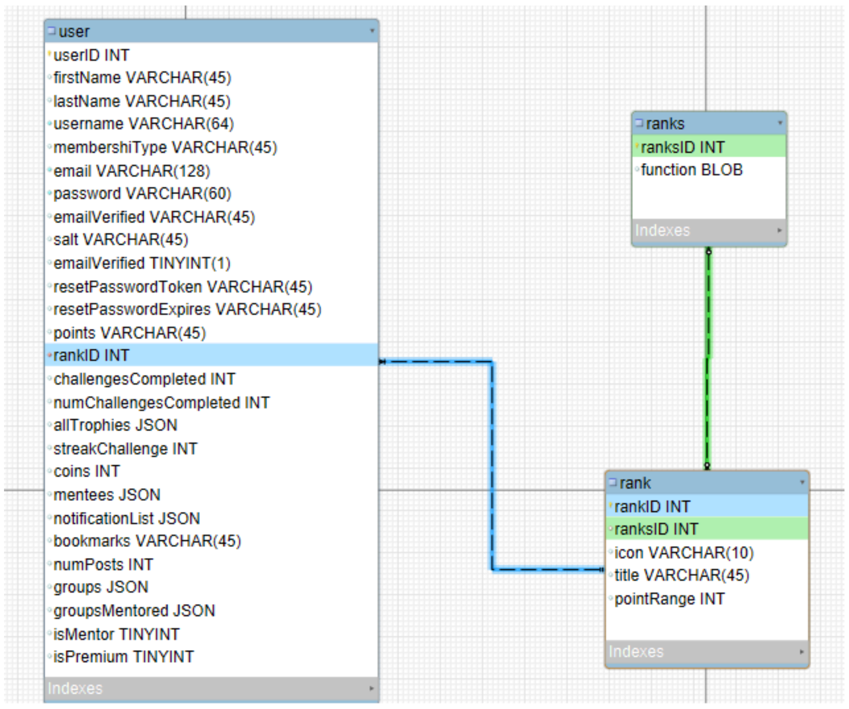
1. Overview



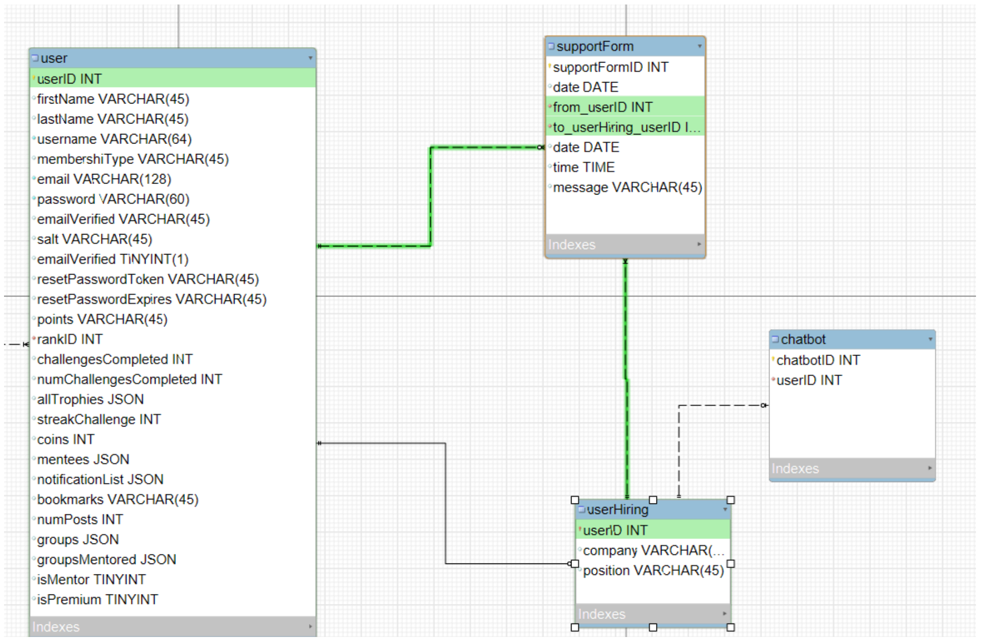
github link:
https://github.com/sfsu-joseo/csc648-848-05-sw-engineering-su24-T1/blob/will-backend-dev/application/server/db/High_level_database_architecture/EERv2.mwbbranch:will-backend-dev

folder: application/server/db/High_level_database_architecture/ EERv2.mwb

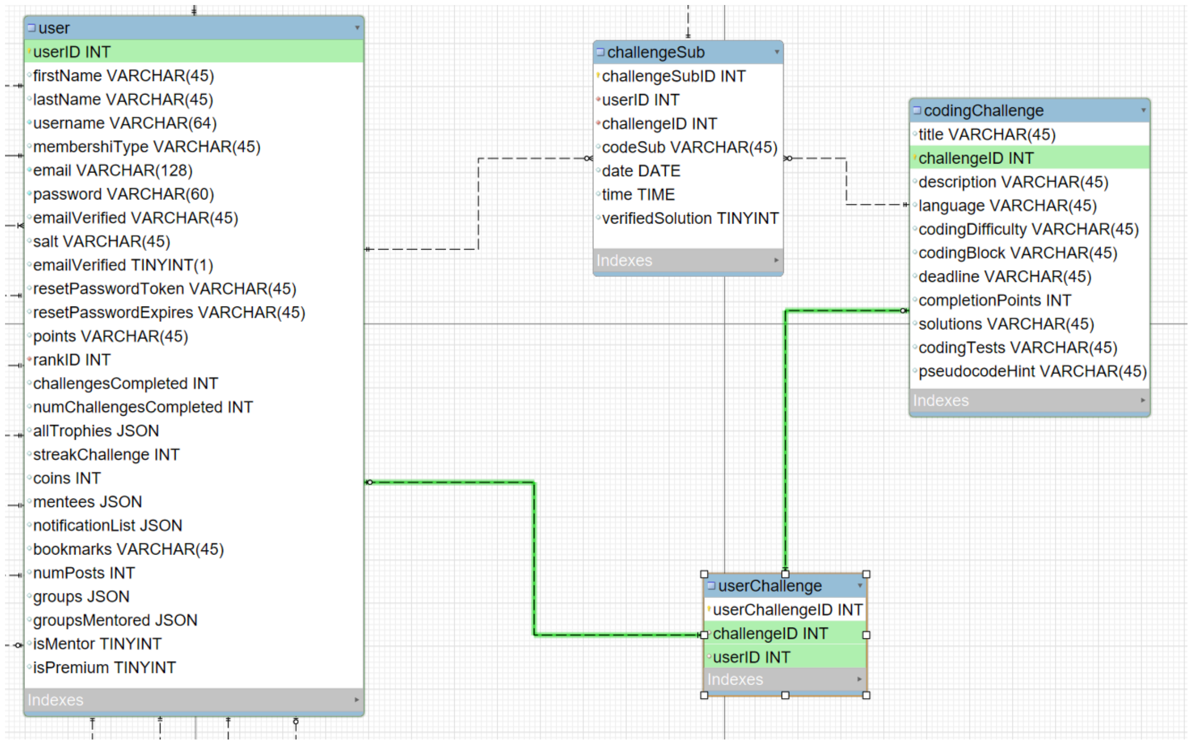
1.1. Ranks and rank



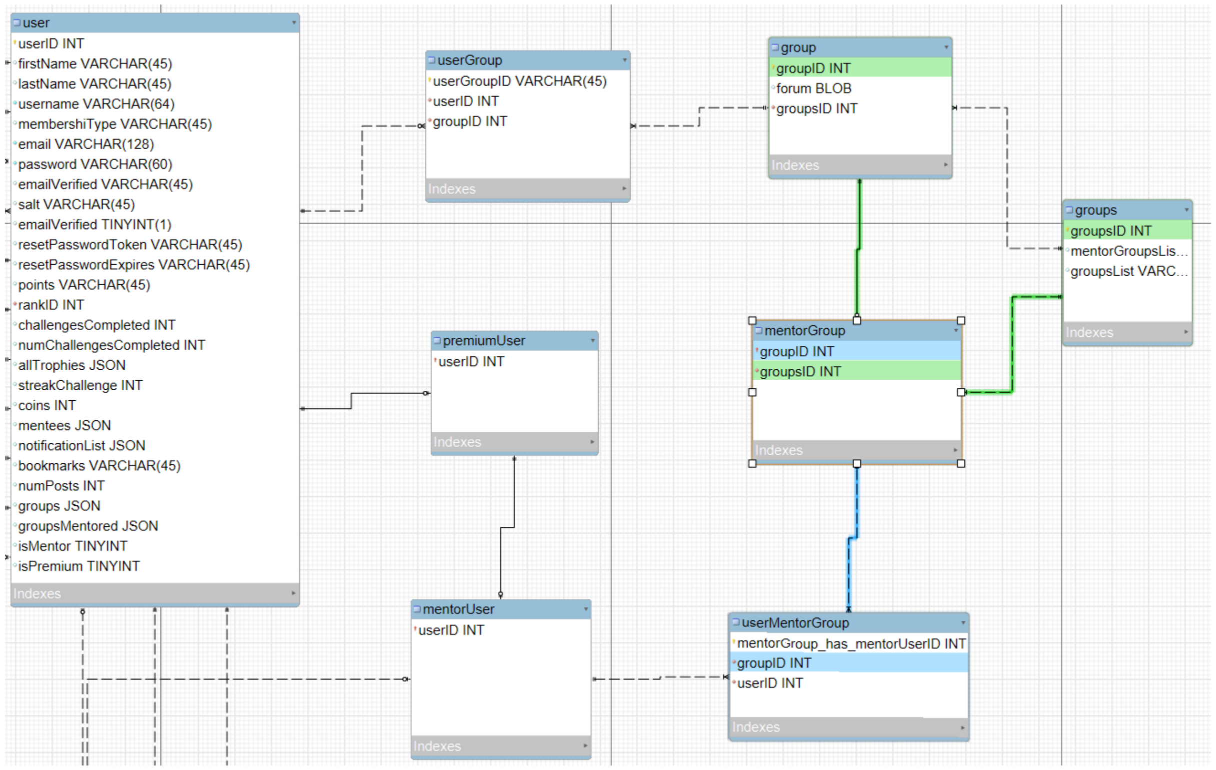
1.2. userHiring and support form



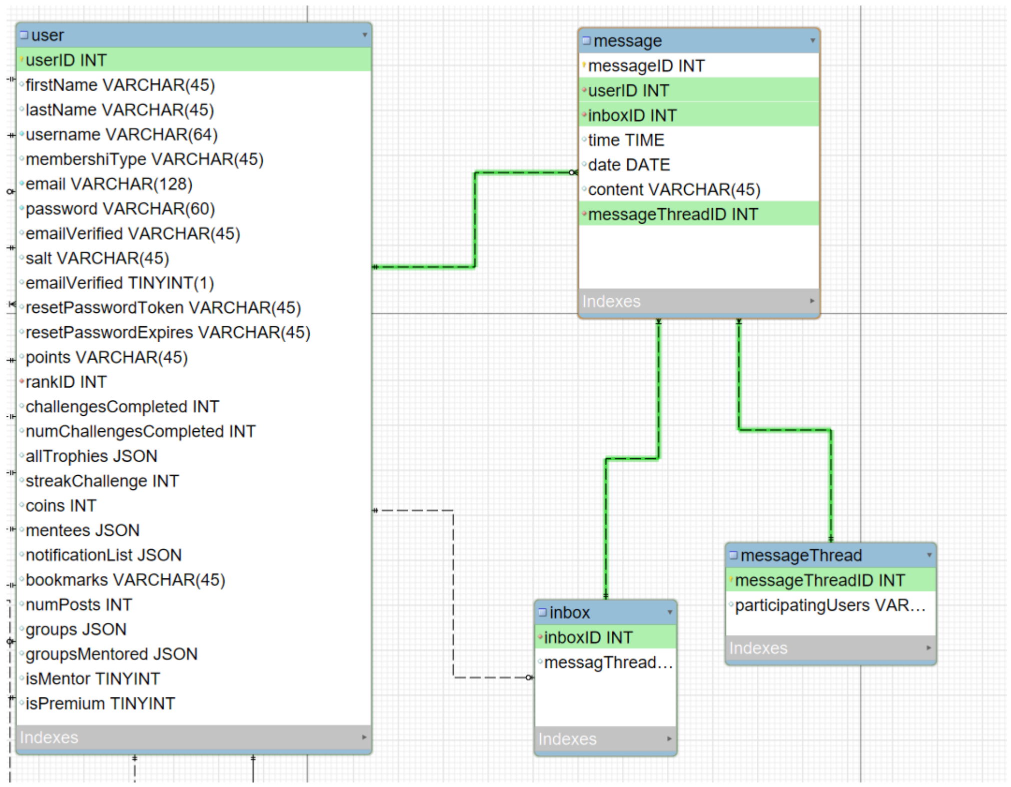
1.3. UserChallenge , codingChallenge and challengeSub



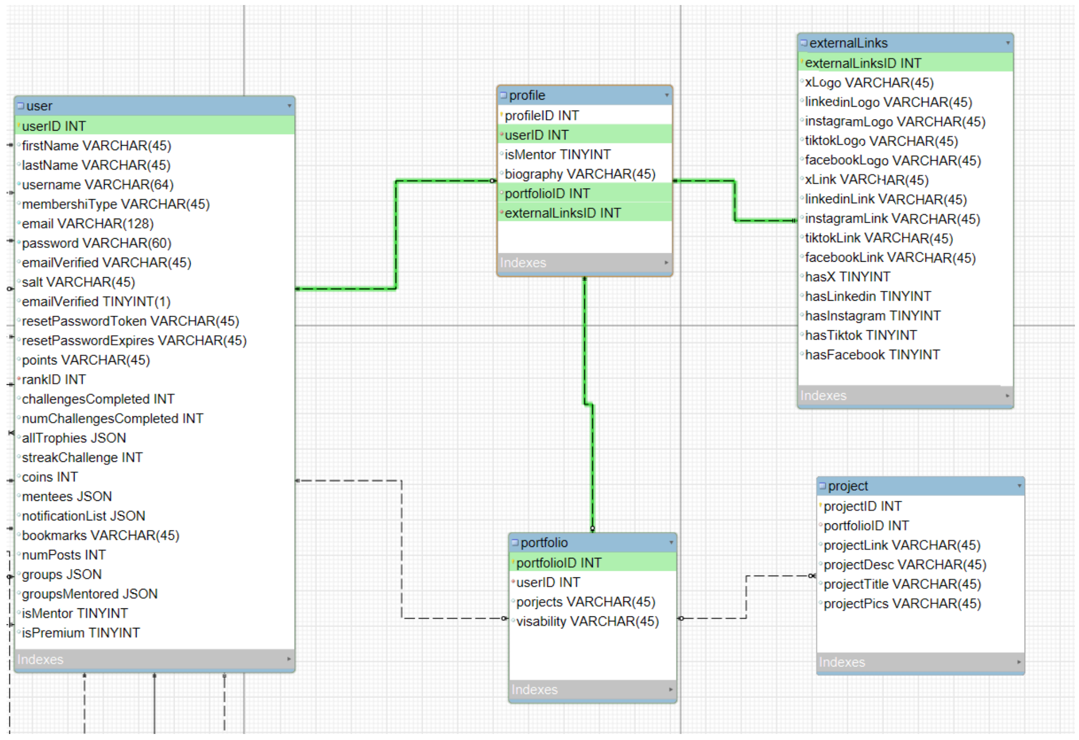
1.4. Group



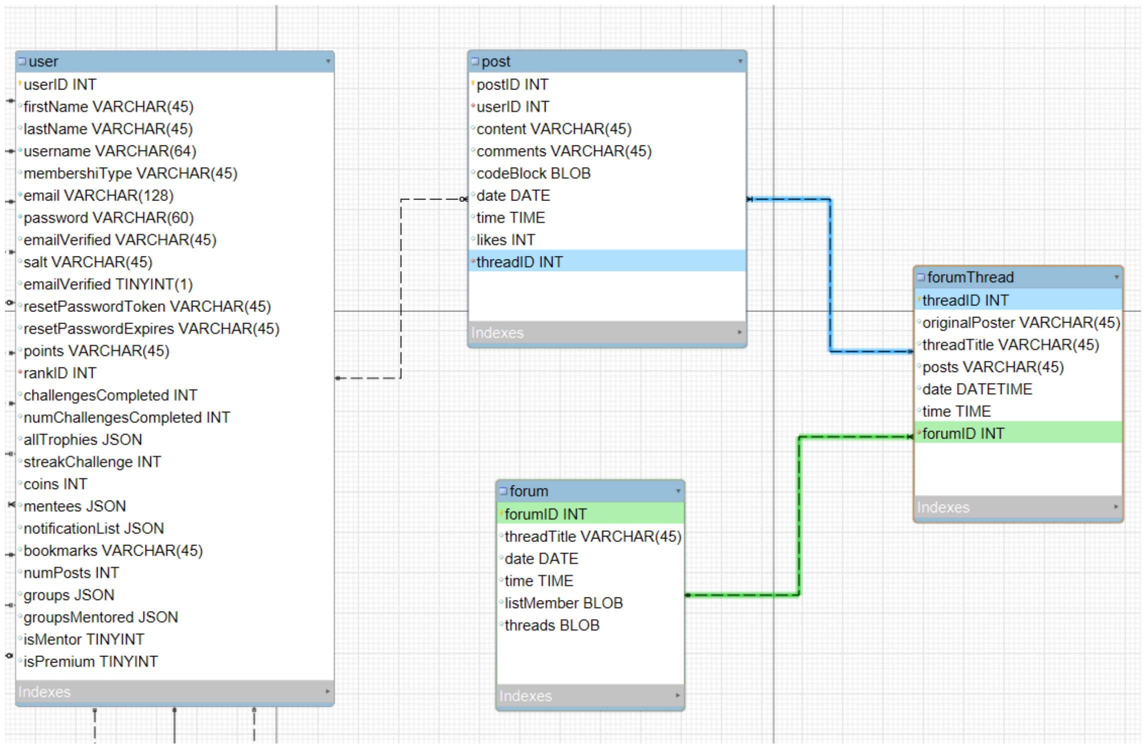
1.5. Message and inbox



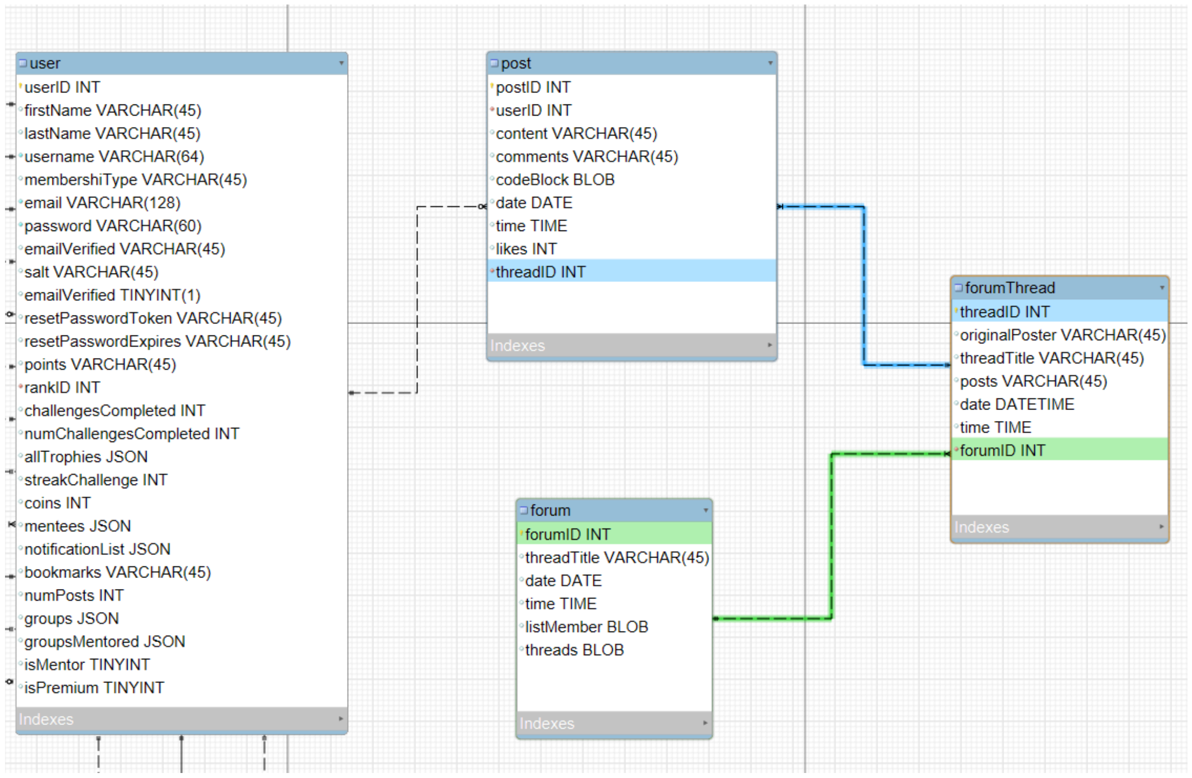
1.6. Profile, portfolio project and external link



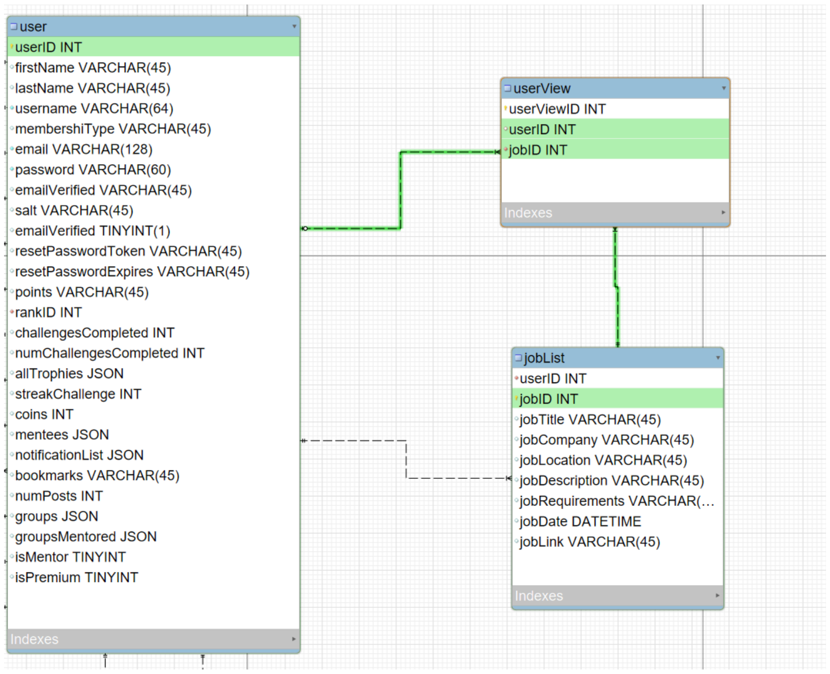
1.7. Post, forumThread,forum



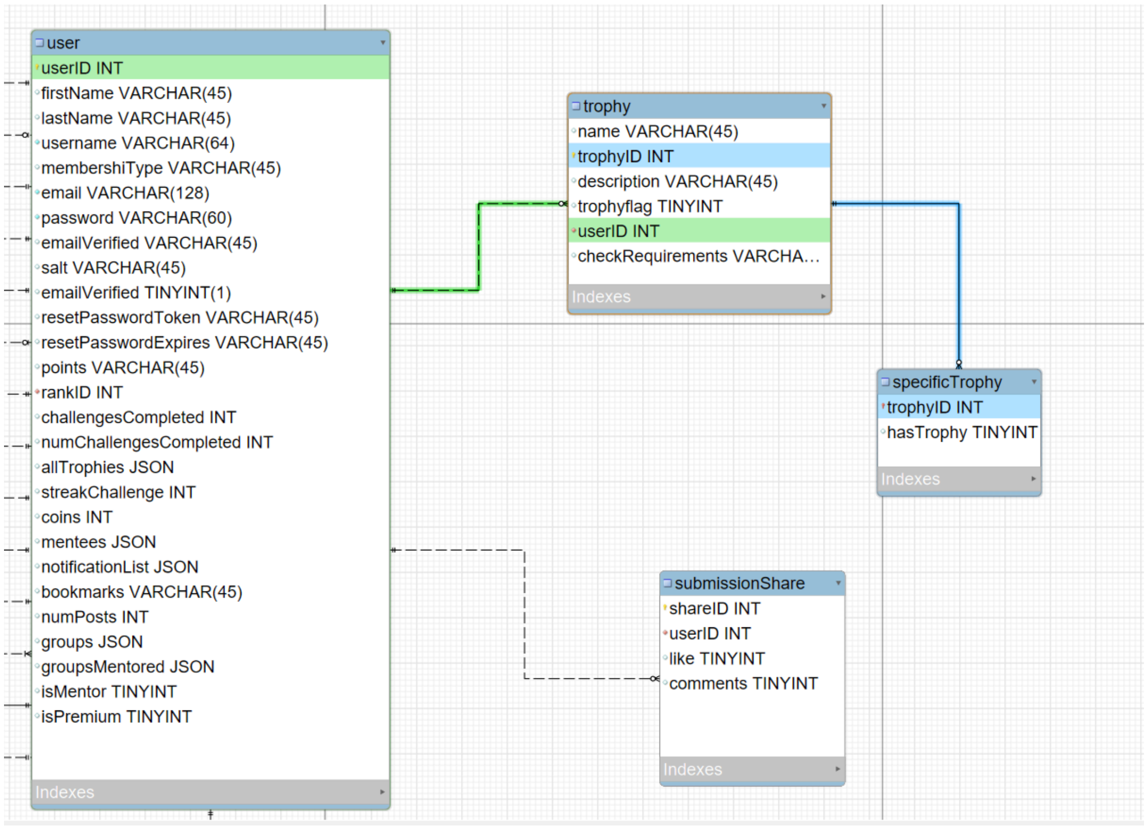
1.8. Notification



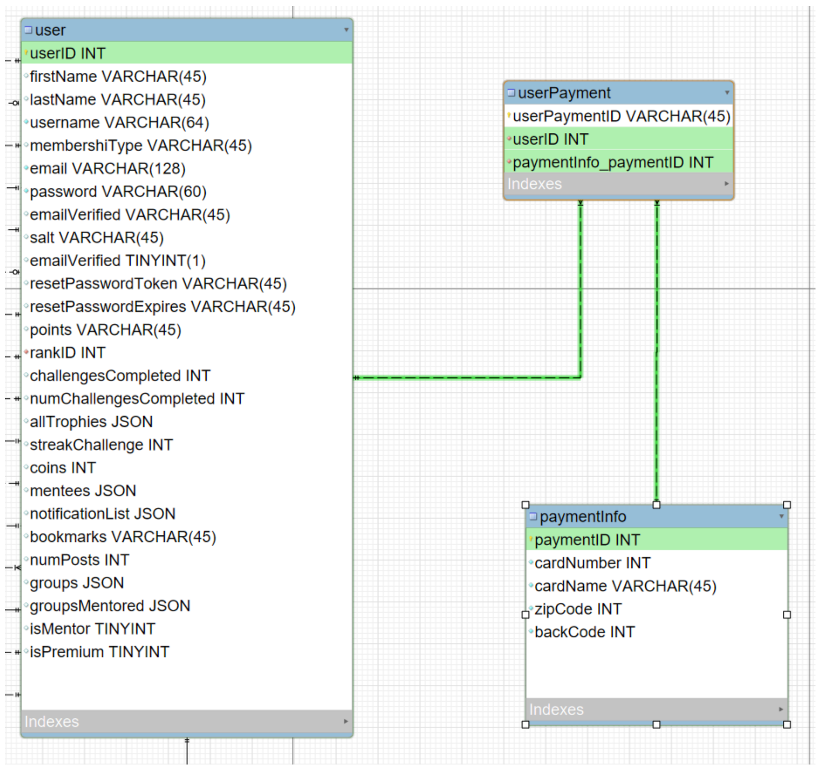
1.9. jobList and userView



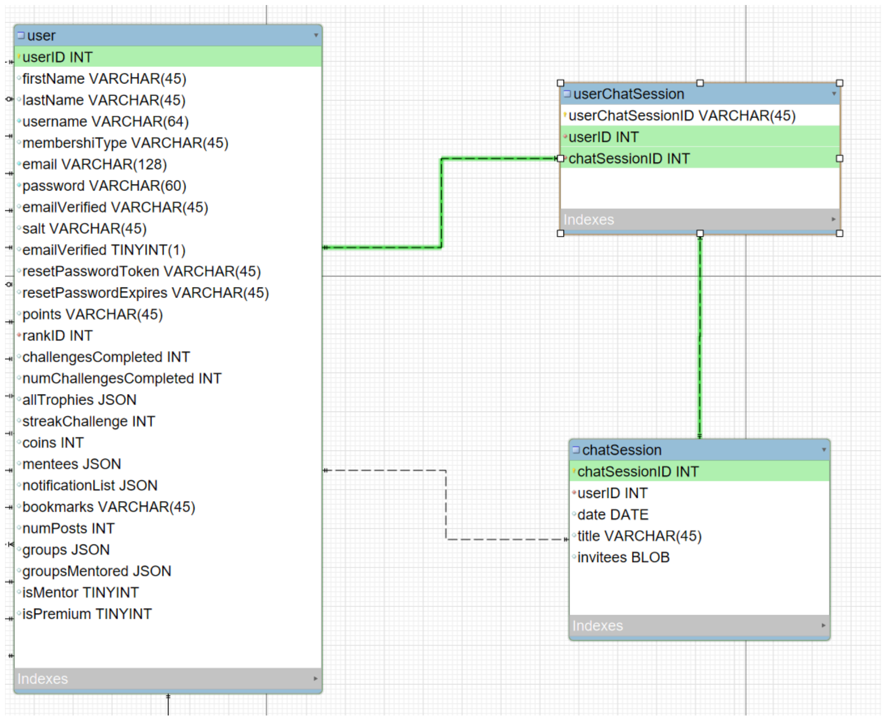
1.10. Trophy and submission share



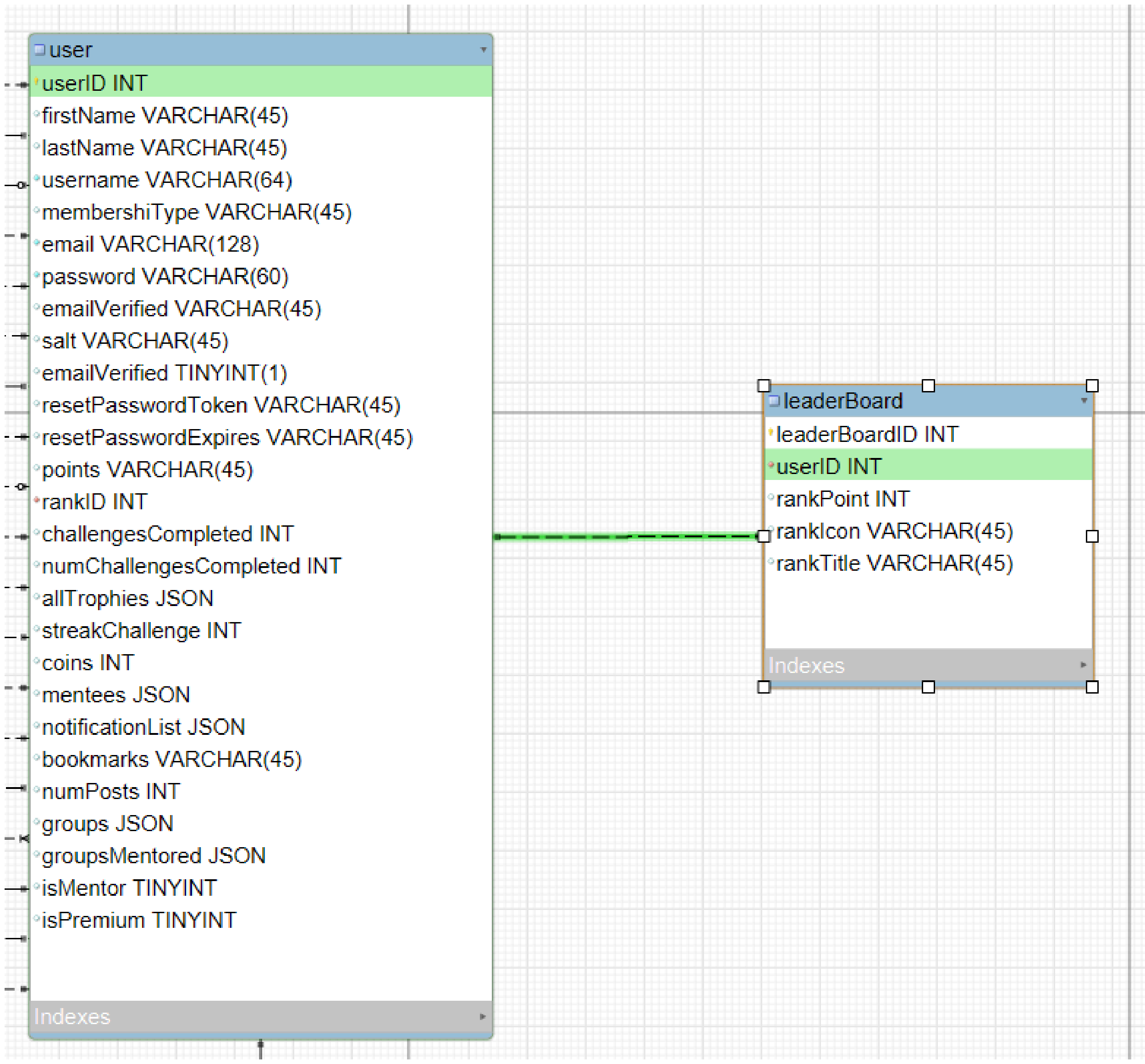
1.11. Payment and userPayment



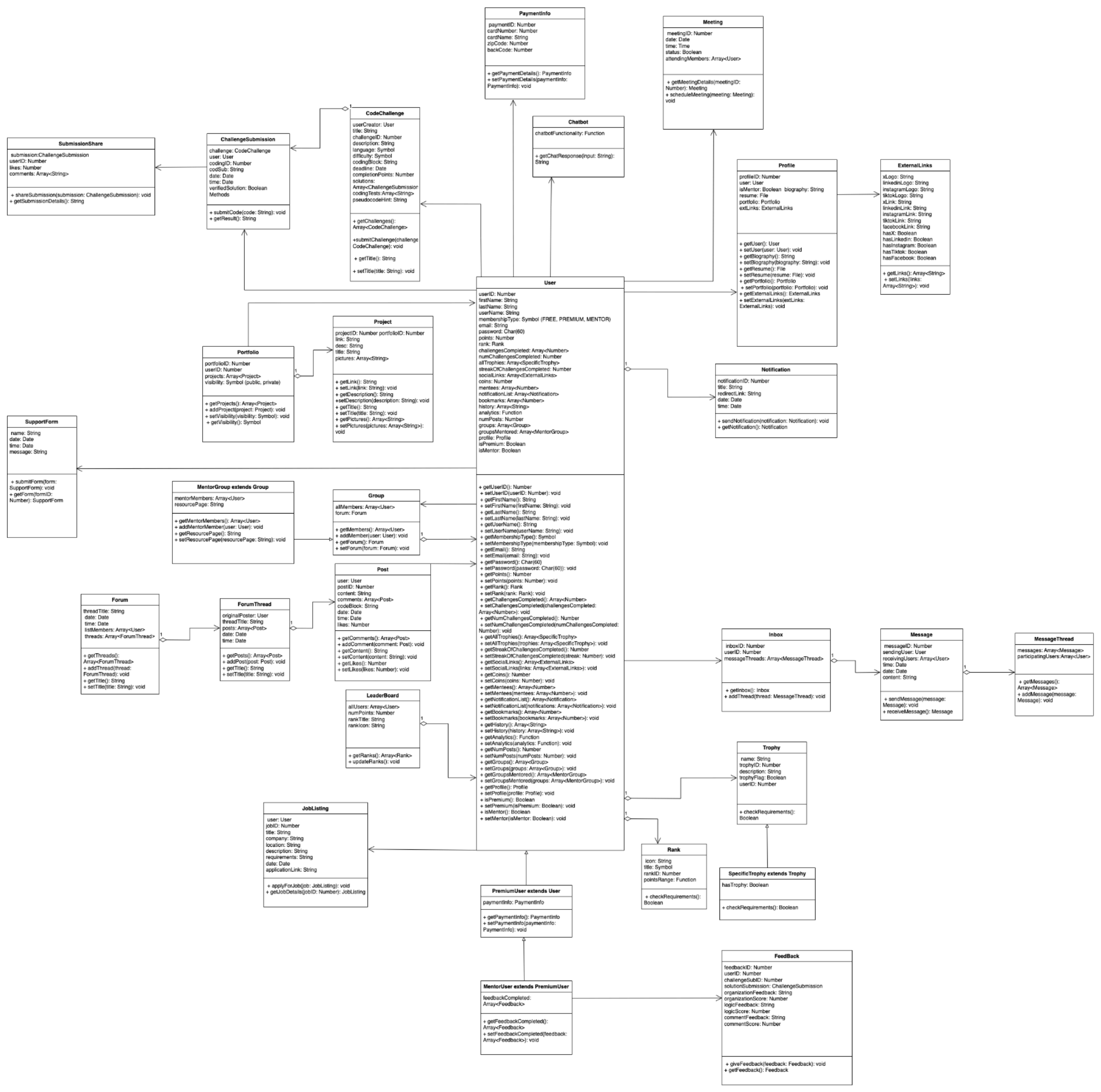
1.12. chatSession



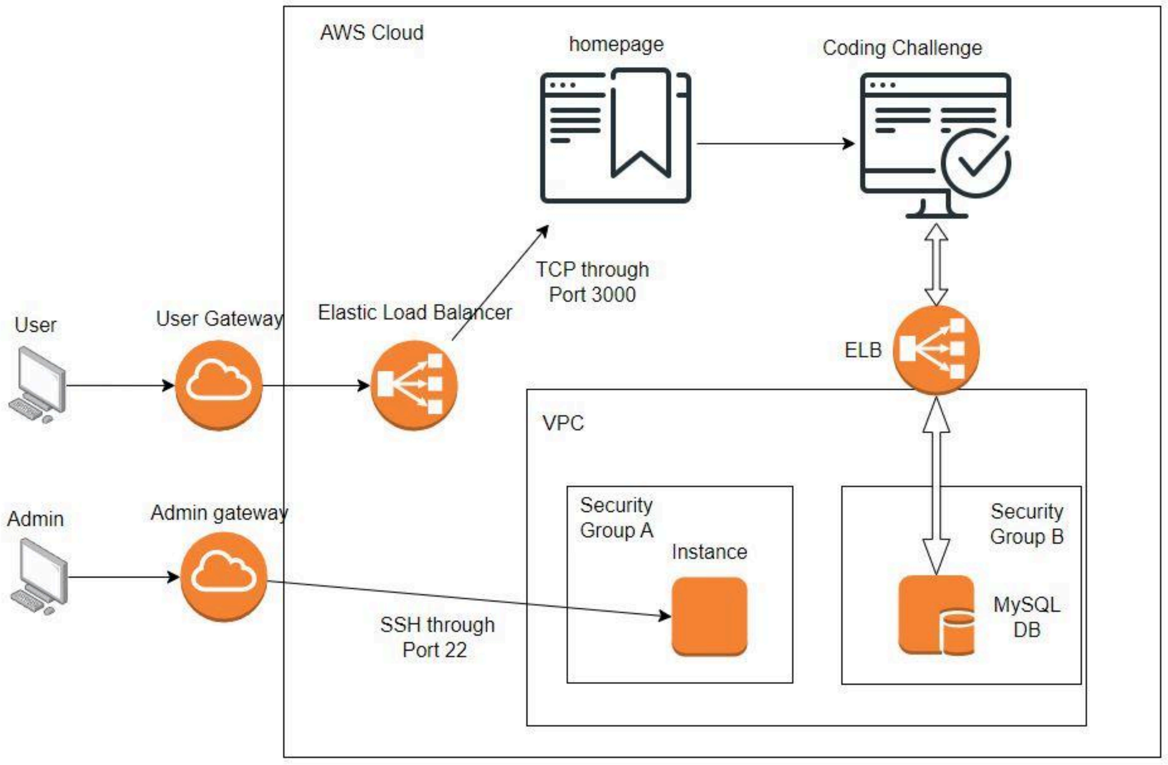
1.13. leaderBoard



5. High Level Diagrams (detailed)
UML:


PaymentInfo
paymentID: Number
cardNumber: Number
cardName: String
zipCode: Number
backCode: Number
+ getPaymentDetails(): PaymentInfo
+ setPaymentDetails(paymentInfo: PaymentInfo): void
Meeting
meetingID: Number
date: Date
time: Time
status: Boolean
attendingMembers: Array<User>
+ getMeetingDetails(meetingID: Number): Meeting
+ scheduleMeeting(meeting: Meeting): void
Chatbot
chatbotFunctionality: Function
+ getChatResponse(input: String): String
CodeChallenge
userCreator: User
title: String
challengeID: Number
description: String
language: Symbol
difficulty: Symbol
codingBlock: String
deadline: Date
completionPoints: Number
solutions: Array<ChallengeSubmission
codingTests:Array<String>
pseudocodeHint: String
+ getChallenges(): Array<CodeChallenge>
+submitChallenge(challenge CodeChallenge): void
+ getTitle(): String
+ setTitle(title: String): void
SubmissionShare
submission:ChallengeSubmission
userID: Number
likes: Number
comments: Array<String>
+ shareSubmission(submission: ChallengeSubmission): void
+ getSubmissionDetails(): String
ChallengeSubmission
challenge: CodeChallenge
user: User
codingID: Number
codSub: String
date: Date
time: Date
verifiedSolution: Boolean
Methods
+ submitCode(code: String): void
+ getResult(): String
Profile
profileID: Number
user: User
isMentor: Boolean biography: String
resume: File
portfolio: Portfolio
extLinks: ExternalLinks
+ getUser(): User
+ setUser(user: User): void
+ getBiography(): String
+ setBiography(biography: String): void
+ getResume(): File
+ setResume(resume: File): void
+ getPortfolio(): Portfolio
+ setPortfolio(portfolio: Portfolio): void
+ getExternalLinks(): ExternalLinks
+ setExternalLinks(extLinks: ExternalLinks): void
ExternalLinks
xLogo: String
linkedinLogo: String
instagramLogo: String
tiktokLogo: String
xLink: String
linkedinLink: String
instagramLink: String
tiktokLink: String
facebookLink: String
hasX: Boolean
hasLinkedin: Boolean
hasInstagram: Boolean
hasTiktok: Boolean
hasFacebook: Boolean
+ getLinks(): Array<String>
+ setLinks(links: Array<String>): void
User
userID: Number
firstName: String
lastName: String
userName: String
membershipType: Symbol (FREE, PREMIUM, MENTOR)
email: String
password: Char(60)
points: Number
rank: Rank
challengesCompleted: Array<Number>
numChallengesCompleted: Number
allTrophies: Array<SpecificTrophy>
streakOfChallengesCompleted: Number
socialLinks: Array<ExternalLinks>
coins: Number
mentees: Array<Number>
notificationList: Array<Notification>
bookmarks: Array<Number>
history: Array<String>
analytics: Function
numPosts: Number
groups: Array<Group>
groupsMentored: Array<MentorGroup>
profile: Profile
isPremium: Boolean
isMentor: Boolean
+ getUserID(): Number
+ setUserID(userID: Number): void
+ getFirstName(): String
+ setFirstName(firstName: String): void
+ getLastName(): String
+ setLastName(lastName: String): void
+ getUserName(): String
+ setUserName(userName: String): void
+ getMembershipType(): Symbol
+ setMembershipType(membershipType: Symbol): void
+ getEmail(): String
+ setEmail(email: String): void
+ getPassword(): Char(60)
+ setPassword(password: Char(60)): void
+ getPoints(): Number
+ setPoints(points: Number): void
+ getRank(): Rank
+ setRank(rank: Rank): void
+ getChallengesCompleted(): Array<Number>
+ setChallengesCompleted(challengesCompleted: Array<Number>): void
+ getNumChallengesCompleted(): Number
+ setNumChallengesCompleted(numChallengesCompleted: Number): void
+ getAllTrophies(): Array<SpecificTrophy>
+ setAllTrophies(trophies: Array<SpecificTrophy>): void
+ getStreakOfChallengesCompleted(): Number
+ setStreakOfChallengesCompleted(streak: Number): void
+ getSocialLinks(): Array<ExternalLinks>
+ setSocialLinks(links: Array<ExternalLinks>): void
+ getCoins(): Number
+ setCoins(coins: Number): void
+ getMentees(): Array<Number>
+ setMentees(mentees: Array<Number>): void
+ getNotificationList(): Array<Notification>
+ setNotificationList(notifications: Array<Notification>): void
+ getBookmarks(): Array<Number>
+ setBookmarks(bookmarks: Array<Number>): void
+ getHistory(): Array<String>
+ setHistory(history: Array<String>): void
+ getAnalytics(): Function
+ setAnalytics(analytics: Function): void
+ getNumPosts(): Number
+ setNumPosts(numPosts: Number): void
+ getGroups(): Array<Group>
+ setGroups(groups: Array<Group>): void
+ getGroupsMentored(): Array<MentorGroup>
+ setGroupsMentored(groups: Array<MentorGroup>): void
+ getProfile(): Profile
+ setProfile(profile: Profile): void
+ isPremium(): Boolean
+ setPremium(isPremium: Boolean): void
+ isMentor(): Boolean
+ setMentor(isMentor: Boolean): void
Project
projectID: Number portfolioID: Number
link: String
desc: String
title: String
pictures: Array<String>
+ getLink(): String
+ setLink(link: String): void
+ getDescription(): String
+setDescription(description: String): void
+ getTitle(): String
+ setTitle(title: String): void
+ getPictures(): Array<String>
+ setPictures(pictures: Array<String>): void
Portfolio
portfolioID: Number
userID: Number
projects: Array<Project>
visibility: Symbol (public, private)
+ getProjects(): Array<Project>
+ addProject(project: Project): void
+ setVisibility(visibility: Symbol): void
+ getVisibility(): Symbol
Notification
notificationID: Number
title: String
redirectLink: String
date: Date
time: Date
+ sendNotification(notification: Notification): void
+ getNotification(): Notification
SupportForm
name: String
date: Date
time: Date
message: String
+ submitForm(form: SupportForm): void
+ getForm(formID: Number): SupportForm
MentorGroup extends Group
mentorMembers: Array<User>
resourcePage: String
+ getMentorMembers(): Array<User>
+ addMentorMember(user: User): void
+ getResourcePage(): String
+ setResourcePage(resourcePage: String): void
Group
allMembers: Array<User>
forum: Forum
+ getMembers(): Array<User>
+ addMember(user: User): void
+ getForum(): Forum
+ setForum(forum: Forum): void
Post
user: User
postID: Number
content: String
comments: Array<Post>
codeBlock: String
date: Date
time: Date
likes: Number
+ getComments(): Array<Post>
+ addComment(comment: Post): void
+ getContent(): String
+ setContent(content: String): void
+ getLikes(): Number
+ setLikes(likes: Number): void
Forum
threadTitle: String
date: Date
time: Date
listMembers: Array<User>
threads: Array<ForumThread>
+ getThreads(): Array<ForumThread>
+ addThread(thread: ForumThread): void
+ getTitle(): String
+ setTitle(title: String): void
ForumThread
originalPoster: User
threadTitle: String
posts: Array<Post>
date: Date
time: Date
+ getPosts(): Array<Post>
+ addPost(post: Post): void
+ getTitle(): String
+ setTitle(title: String): void
Inbox
inboxID: Number
userID: Number
messageThreads: Array<MessageThread>
+ getInbox(): Inbox
+ addThread(thread: MessageThread): void
Message
messageID: Number
sendingUser: User
receivingUsers: Array<User>
time: Date
date: Date
content: String
+ sendMessage(message: Message): void
+ receiveMessage(): Message
MessageThread
messages: Array<Message>
participatingUsers:Array<User>
+ getMessages(): Array<Message>
+ addMessage(message: Message): void
LeaderBoard
allUsers: Array<User>
numPoints: Number
rankTitle: String
rankIcon: String
+ getRanks(): Array<Rank>
+ updateRanks(): void
Trophy
name: String
trophyID: Number
description: String
trophyFlag: Boolean
userID: Number
+ checkRequirements(): Boolean
JobListing
user: User
jobID: Number
title: String
company: String
location: String
description: String
requirements: String
date: Date
applicationLink: String
+ applyForJob(job: JobListing): void
+ getJobDetails(jobID: Number): JobListing
Rank
icon: String
title: Symbol
rankID: Number
pointsRange: Function
+ checkRequirements(): Boolean
SpecificTrophy extends Trophy
hasTrophy: Boolean
+ checkRequirements(): Boolean
PremiumUser extends User
paymentInfo: PaymentInfo
+ getPaymentInfo(): PaymentInfo
+ setPaymentInfo(paymentInfo: PaymentInfo): void
MentorUser extends PremiumUser
feedbackCompleted: Array<Feedback>
+ getFeedbackCompleted(): Array<Feedback>
+ setFeedbackCompleted(feedback: Array<Feedback>): void
FeedBack
feedbackID: Number
userID: Number
challengeSubID: Number
solutionSubmission: ChallengeSubmission
organizationFeedback: String
organizationScore: Number
logicFeedback: String
logicScore: Number
commentFeedback: String
commentScore: Number
+ giveFeedback(feedback: Feedback): void
+ getFeedback(): Feedback
1

Network Diagram:



6. List of Contributions in this milestone (detailed including contributions to the horizontal prototype)

Max Shigeyoshi: (6/10)

- Corrected search functionality
- Backend-frontend integration
- Milestone 3 document creation
- Merging branches to integration then to master.

Aaron Rayray(7/10):

- Updated css for all pages
- HTML Files : Splash, Workspace, Java best practices, Java, Javascript, Store, Room1-15, React getting started
- Header configurations for all pages
- Footer configuration for all pages
- Creating scripts for pages that needed redirecting

Noah Hai: (7/10)

- Created mentor pages,
- Created premium page
- Created book a mentor page
- Created find a mentor page

Shez Rahman: (9/10)

- Backend-frontend integration
- Corrected sign in and sign up features
- Corrected email verification
- Corrected forgot password function
- Troubleshoot all sections of code

Ghadeer Al-Badani: (8/10)

- Refactored coding style
- Troubleshoot backend sections of code
- Assisted in search functionality
- Frontend backend integration
- Created commenting standard

Majd Alnajjar: (8/10)

- Created a leaderboard where users can view other's accomplishments with the ability to open profiles
- Created a job posting page where companies post hiring opportunities
- Created an application page where users can apply for said company
- Created profile pages for 8 mock users on the leaderboards
- Added a blank profile picture to profile.html
- HTML Files: alice.html, bob.html, dave.html, carol,html, frank.html, eve.html, john.html, lenny.html, leaderboards.html, jobs.html, application.html
- Wireframe for toby

William Pan: (8/10)

- Managed Mysql database
- Updated models to create database schemas
-

Phillip Ma: (7/10)

- Css style changes for all pages
- Search function integration
- Profile, forgot password, forgot password verification, homepage, inbox, explore, challenges
- Wireframes
- Frontend backend integration