Team Members

- Nicholas Matsumoto
- Brian Smith

Project Name

CS340_DBProject – User Authentication Prototype Group

Project Overview

Purpose

Our project simulates the behaviors and storage patterns of Single Sign On Provider. Henceforth referred to as SSO or Single Sign On. In accordance with the requirements laid out in the following documents and the requirements of Oregon State University E-Campus CS340 "Introduction to Databases" course.

- CS340 Project Guide
  - https://canvas.oregonstate.edu/courses/1825733/pages/cs340-project-guide?module_item_id=20221718
- Project Step 1 – Project Proposal and Outline
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147636?module_item_id=20221732

as well as the following documents when they are made available by the classroom environment.

- Project Step 2 Draft Version
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147637
- Project Step 2 Review
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147639
- Project Step 2 Final Version
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147638
- Project Step 3 Review
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147642
- Project Step 3 Final Version
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147641
- Project Step 4 Draft Version
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147643
- Project Step 5 Draft Version
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147645
- Project Step 5 Review
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147646
- Project Step 6 Draft Version
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147647
- Project Step 6 Review
  - https://canvas.oregonstate.edu/courses/1825733/assignments/8147648

- Project Step 7
    - https://canvas.oregonstate.edu/courses/1825733/assignments/8147649

Problem Statement

Our application will implement a solution to the issues of credential reuse, validation, authority, revocation, and interoperability most often solved by SSO infrastructure. Our application due to additional constraints posed by the CS340 project will vary from traditional SSO infrastructure both in scope and execution. These variances will be applied to ensure compliance with grading rubric requirements.

Measures of Success and Operational Objectives

Our project will be measured a success upon the suitable completion of the listed objectives.

- Successful execution of all grading, project guidance, university, and classroom constraints.
- Implementation of a UI that allows for the following operations
    - CRUD operations for a user
    - CRUD operations for a credential
    - CRUD operations for a role
    - CRUD operations for a session
    - Sign on into our application
    - Log off from our application
- Implementation of an API that allows for the following operations
    - CRUD operations for a user
    - CRUD operations for a credential
    - CRUD operations for a role
    - CRUD operations for a session
    - Validation of a session
    - Creation of a session token for a successful logon
    - Revocation of a session token

Database Design and Operational Diagrams

Our schema implements four data tables and two *join* tables to facilitate a pair of *many to many* relationships within our data structure. The database entities and properties are currently designed as follows.

User

The user table contains information about registered users within our database. This table contains the following fields

- ID (PK, NOT NULL)
    - The ID field is both the USERID number for referencing a user in our application and in pivot tables as well as the Primary Key for this table.
- Username (varchar, NOT NULL, UNIQUE)

    - The Username field stores the username for as registered user and it used as part our our application logic and identifying specific users during requests for validation and authorization purposes.
- Created_at (datetime, NOT NULL)
    - The Created_at field is a timestamp to identify when as user was created. This value was generated specifically for use in interesting statistical work should we desire in the future as well as provide some *flair* in our data design.
- Email (varchar)
    - The email field store the users preferred email address. This value is generated as *flair* in our data design as well as *realism* in our simulated application.

Session

The session table is used to track ongoing user sessions. This table is referenced to determine if a user has a valid session on-going when attempting to preform an operation, store generated session tokens, maintain statistical information about a session, and provide historical records to session activity with in our application. This table contains the following fields.

- ID (pk, auto-increment, NOT NULL)
    - Contains the *Session ID* used to identify unique sessions with in our database.
    - Acts as the tables PK
- User_id (fk -> database.user)
    - Foreign Key
    - Links a session to a specific requesting user
- Token (varchar)
    - Stores the session key for authenticating a specific user active session for operations.
- Exp_date (timestamp)
    - Stores the time the session becomes invalid
- User_req_date (timestamp)
    - Stores the time the users client believes it requested the session.
- Created_at (timestamp)
    - Stores the time our service generated the session for the user.

Role

The role table contains the mapping for the kind of users our application allows to exists and the permission they are allowed to execute. This table contains the following fields.

- ID (pk)
    - Acts as the table PK
    - Provides the id for a role
- Logon (bool)
    - Reports if the role is allowed to logon an generate a session token
- Update (bool)
    - Reports if the role is allowed to update their own user information
- Auth (bool)
    - Reports if the role is allowed to Authenticate with the service using a session token

- Validate (bool)
  - Reports if the role is allowed to request validation of a session
- Super (bool)
  - Reports if the role is a super user
- Special (bool)
  - Reports if the role has special permissions
  - Reserved for future use
- Role_title (varchar)
  - Store the human readable name of the role.

Credential

The credential table contains the valid credentials for a user so a users logon requests can be serviced. This table contains the following fields

- ID (pk)
  - The ID acts as the table PK
  - Acts a credential id
- Hash (varchar)
  - Contains the users password hash
- Salt (varchar)
  - Contains the users unique database password salt to hinder rainbow table operations
- Exp_date (timestamp)
  - Contains the credentials expiration date
- Created_date (timestamp)
  - Contains the credentials creation date
- Enabled (bool)
  - Reports if the credential is active and can be used for authenticating a user during log on.

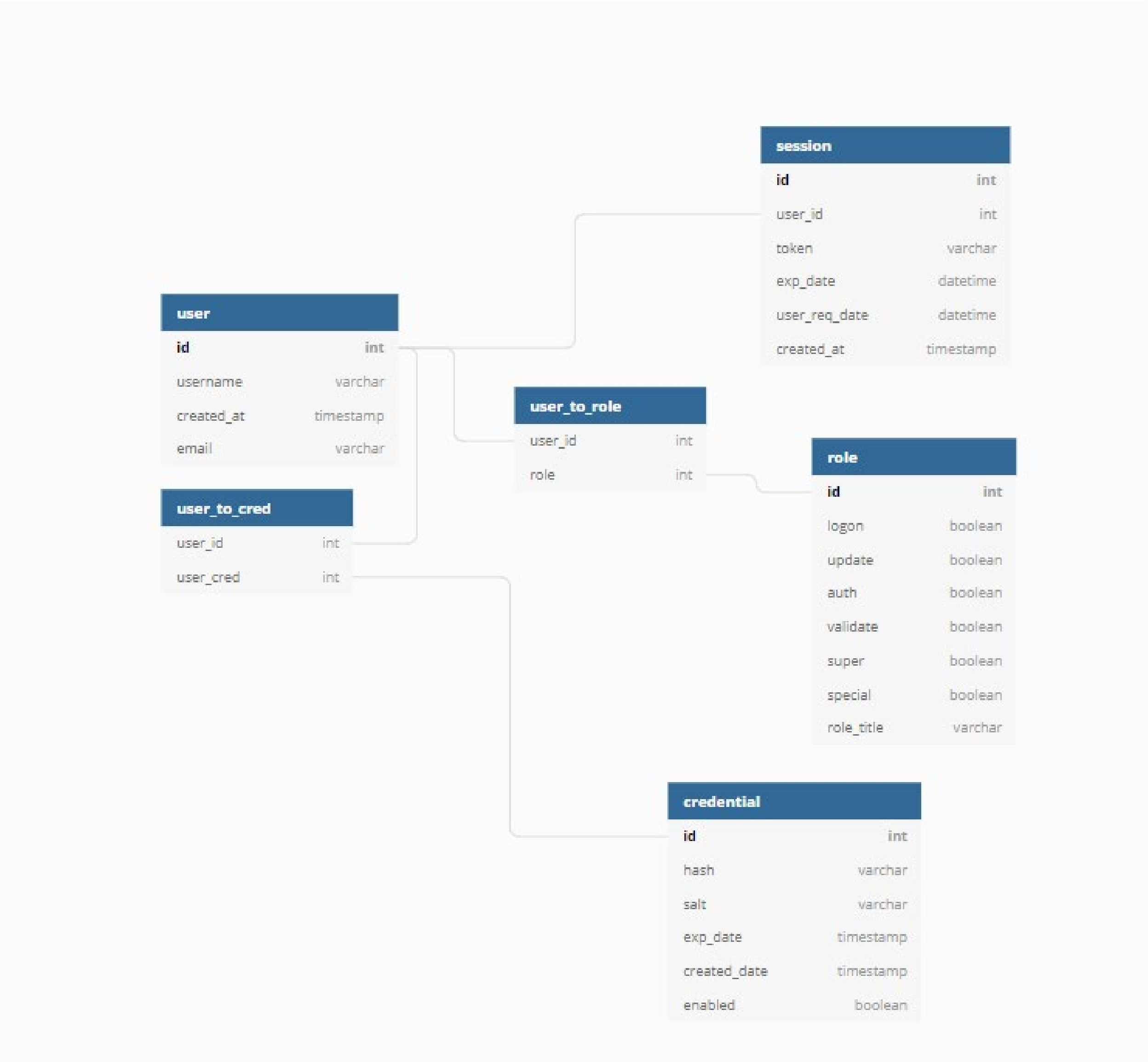
User_to_Role

The user_to_role table is used to map as users to authorized roles for a given user. This table contains only foreign keys of the user and role tables to act as a map.

User_to_Credential

The user_to_credential table is used to map users to credentials allowed to validate their log on attempts. This table contains only foreign keys of the user and credentials tables to act as a map.

Database Diagram Image



session	
id	int
user_id	int
token	varchar
exp_date	datetime
user_req_date	datetime
created_at	timestamp

user	
id	int
username	varchar
created_at	timestamp
email	varchar

user_to_role	
user_id	int
role	int

role	
id	int
logon	boolean
update	boolean
auth	boolean
validate	boolean
super	boolean
special	boolean
role_title	varchar

user_to_cred	
user_id	int
user_cred	int

credential	
id	int
hash	varchar
salt	varchar
exp_date	timestamp
created_date	timestamp
enabled	boolean

Conclusion

This document provides the initial draft of the requested project specifications for our group. Additional questions may be presented to ether group members. Additional revisions may follow.