Project team # - <Project Title/Name>

Team Members: Tommy, Lance

PROJECT PROPOSAL

Content, Scope and Objectives

OAuth systems must manage information about tokens handed to the client by the provider (ex: Google). Such as access/refresh token, client ID/Secret, Token binding (roles), and expiration times.

PROJECT ENVIRONMENT

Authentication provides system database for managing client and session information. There is a master table which holds information about the system owner. There is another table designed for the system owner to distribute their own keys with certain attributes. This allows for the owner to distribute authentication without ever giving admin access to anyone but themselves. This also allows the OAuth system managers to have a simple 2 table setup for many admins and users with each client being associated with a Master Entity and having their own special access token.

HIGH LEVEL REQUIREMENTS

Initial user roles

User Role	Description
Session/Master	Any system requesting token to generate user tokens from
Master/Session Users	Stores access tokens, roles, and expiration from the master/session token

Initial user story descriptions

Story ID	Story description
Session/	I want to allow users to login once with username and

Master Developer	password and provide them with a valid token. This token can be used to login on later requests and expires at a certain time past the issue date. This allows users to not have to re-login on each request and allows the system developer to distribute different types of tokens to their users.
Master/Session Users	Logging into a website once and then being able to come back and remember that I'm logged in is amazing. My friend also uses a special role that allowed them to do certain things which were awesome to see in action!

HIGH LEVEL CONCEPTUAL DESIGN

Entities:

Session/Master:

ID (primary key for database)

Entity ID (name of the owner of the master token)

Master Token (Used to validate other users within that system)

Session/Master Users:

Client ID (generated from the master token is the same as a username)

Access Token/Secret

(Generated from the system when instantiating a new client to authenticate on their application/system)

Role (Role of the users, roles are determined by system manager)

Expires In

(How long a token is valid for before a user is required to relog in again)

Relationships:

Session/Master:

Master Token references Entity ID (one to one)

Session/Master Users:

Client ID references Master Token (many to one)

Sprint 1

REQUIREMENTS

Refine the user stories that you made in previous sprint. List your updated user stories and any notes you wish to include in decreasing order of priority and highlight the stories chosen for Sprint 1. *There is no need to show your story refinement process - just the list of updated stories suffices.* Use the format shown below.

Story ID	Story description
US1	As a <role>, I want to <need/feature> so that <reason/benefit>
...	...

CONCEPTUAL DESIGN

Include your detailed conceptual design here. Use the format shown below.

Entity: **Session/Master**
Attributes:
ID (primary key)
Entity ID
Master Token
Examples:
attr1_b [composite]
part_1
part_2

Entity: **Session/Master Users**
Attributes:
Client ID [derived]
Access Token/Secret
Role

Expiry
Examples:
attr2_a
attr2_b [multi-valued]
attr2_c [derived]

Relationship: **Master/Session** manages **Master/Session Users**
Cardinality: One to Many
Participation:
(A master could have no users but there can be no users without a master)
Entity1 has total participation
Entity2 has partial participation

LOGICAL DESIGN

Include your logical design here. Use the format shown below.

Table: **Admin**
Columns:
ID
Entity_ID
Master_Token

Examples:
pk_1
column_1a
column_1b

Justification (if needed)

Table: **Users**
Columns:
Client_ID
[foreign key; references **Master_Token** of **Admin**]
Access_Token
Role
Expiry

Examples:
pk_2
column_2a
column_2b [foreign key; references **pk_1** of **Table1**]

The client ID's will contain information from the master token therefore each client will have an identifier being its respective master token. It must be unique as two clients cannot have the same name as they may be issues the same token which is an issue.

SQL QUERIES

List at least **three** SQL queries that perform data retrievals relevant to the features chosen in the current sprint. For each query, paste a **screenshot** of the output, as shown through database management tool.

Sprint 2

REQUIREMENTS

Refine the user stories that you made in previous sprint. List your updated user stories in decreasing order of priority. Highlight the stories for which database design was completed in Sprint 1 in one color. Highlight the updated/new stories chosen for Sprint 2 in a different color. *There is no need to explicitly show your story refinement process.* Use the format shown below.

Story ID	**Story description**
US1	As a <role>, I want to <need/feature> so that <reason/benefit>
...	...

CONCEPTUAL DESIGN

Include your complete updated conceptual design here. Use the format shown below.

Entity: **Entity1**
Attributes:
- attr1_a
- attr1_b [composite]
    - part_1
    - part_2

Entity: **Entity2**
Attributes:
- attr2_a
- attr2_b [multi-valued]
- attr2_c [derived]

Relationship: **Entity1** relationship-phrase **Entity2**
Cardinality: <One/Many> to <One/Many>
Participation:
- Entity1 has <partial/total> participation
- Entity2 has <partial/total> participation

LOGICAL DESIGN WITH NORMAL FORM IDENTIFICATION

Include your complete updated logical design here. Use the format shown below.

Table: **Table1**
Columns:
pk_1
column_1a
column_1b

Justification of primary key (if needed)

Highest normalization level: <1NF/2NF/3NF/BCNF>
Justification (if below BCNF):

Table: **Table2**
Columns:
pk_2
column_2a
column_2b [foreign key; references **pk_1** of **Table1**]

Justification of primary key (if needed)

Highest normalization level: <1NF/2NF/3NF/BCNF>
Justification (if below BCNF):

SQL QUERIES

Refine your SQL queries that you designed in the previous sprint if in need. List at least **three** SQL queries that perform data retrievals relevant to the features chosen in the current sprint. For each query, paste a **screenshot** of the output, as shown through your user interface.

Sprint 3

REQUIREMENTS

Refine the user stories that you made in previous sprint. List your updated user stories in decreasing order of priority. Highlight the stories that were completed in Sprint 1 in one color. Highlight the stories that were completed in Sprint 2 in a different color. Highlight the updated/new stories chosen for Sprint 3, if any, in a third color. *There is no need to explicitly show your story refinement process.* Use the format shown below.

Story ID	**Story description**
US1	As a <role>, I want to <need/feature> so that <reason/benefit>
…	…

CONCEPTUAL DESIGN

Include your complete updated conceptual design here. Use the format shown below.

Entity: **Entity1**
Attributes:
    attr1_a
    attr1_b [composite]
        part_1
        part_2

Entity: **Entity2**
Attributes:
    attr2_a
    attr2_b [multi-valued]
    attr2_c [derived]

Relationship: **Entity1** relationship-phrase **Entity2**
Cardinality: <One/Many> to <One/Many>
Participation:
    Entity1 has <partial/total> participation

Entity2 has <partial/total> participation

LOGICAL DESIGN WITH HIGHEST NORMAL FORMS AND INDEXES

Include your complete updated logical design here. Use the format shown below.

Table: **Table1**
Columns:
pk_1
column_1a
column_1b

Justification of primary key (if needed)

Highest normalization level: <1NF/2NF/3NF/BCNF>
Justification (if below BCNF):

Indexes:
Index #: <type (clustered/non-clustered)>
Columns: <ordered list of columns forming the index>
Justification:

Table: **Table2**
Columns:
pk_2
column_2a
column_2b [foreign key; references **pk_1** of **Table1**]

Justification of primary key (if needed)

Highest normalization level: <1NF/2NF/3NF/BCNF>
Justification (if below BCNF):

Indexes:
Index #: <type (clustered/non-clustered)>
Columns: <ordered list of columns forming the index>
Justification:

VIEWS AND STORED PROGRAMS

List the views relevant to your application here. Use the format specified below.

View: <name of view>
Goal: <1-2 sentence description of what the view contains and what its purpose is (e.g., why and what user(s) would use it, etc.)>

List the stored programs relevant to your application thus far here. Use the format specified below for the different kinds of stored programs. Note: if you do not have a particular type of stored program in your application, just leave that part out.

Stored procedure: <name of procedure>
Parameters: <list of parameters, specifying IN/OUT/INOUT for each>
Goal: <1-2 sentence description of what the stored procedure does>

Stored function: <name of function>
Parameters: <list of parameters>
Goal: <1-2 sentence description of what the stored function does and what it returns>

Trigger: <type of trigger> on <table name>
Goal: <1-2 sentence description of what the trigger does>

Event: <type of event>
Goal: <1-2 sentence description of what the event does>