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 scope, access/refresh token, client ID/Secret, Token binding (user, client, or session), and expiration times.

## PROJECT ENVIRONMENT

Authentication provides system database for managing client and session information. Clients (Users) for different applications will be provided with an auth token and refresh token for that website. The website owner sets up the configuration settings. It also has a session/master token system the same as the client, except it has an extra field to validate tokens with a key. Therefore, an owner can distribute authentication to other users with different roles.

## HIGH LEVEL REQUIREMENTS

### Initial user roles

|  |  |
| --- | --- |
| **User Role** | **Description** |
| Client | Any client requesting access to a session from Google |
| Session/Master | Any system requesting token access for prolonged or intermittent use |

### Initial user story descriptions

|  |  |
| --- | --- |
| **Story ID** | **Story description** |
| Client | As a client I want to provide secure access to my website without managing passwords, usernames, and other information and only must manage keys so that I can increase time producing features rather than focusing on non-user-facing features like authentication while having it secure as possible. |
| Session/Master | As a system owner I want to provide a way to login without managing admin and users. This increases the number of people that can login with other providers as well as increases the security of my application by handing it off to a larger team. They manage who are users and admins with keys and only admins can do the things I want them to do. |

## HIGH LEVEL CONCEPTUAL DESIGN

Entities:

Client Tokens

Session/Master Tokens

…

Relationships:

Entity 1 <relationship phrase> Entity 2

# 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: **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

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

Table: **Table1**

Columns:

pk\_1

column\_1a

column\_1b

*Justification (if needed)*

Table: **Table2**

Columns:

pk\_2

column\_2a

column\_2b [foreign key; references **pk\_1** of **Table1**]

*Justification (if needed)*

## 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>