

Table of Contents

[Table of Contents](#)

[Task Decomposition & SQL Statements](#)

[Login](#)

[View Home Screen](#)

[Add Corkboard](#)

[View Corkboard](#)

[Add Pushpin to Corkboard](#)

[Follow User](#)

[Watch Corkboard](#)

[View Pushpin](#)

[Comment on Pushpin](#)

[Like/Unlike Pushpin](#)

[Search for Pushpin](#)

[View Popular Tags](#)

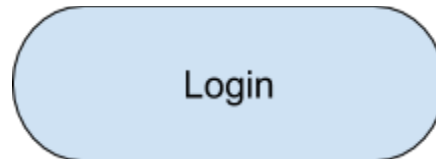
[View Popular Sites](#)

[View Corkboard Statistics](#)

[Table of Contents](#)

Abstract Code & SQL Statements

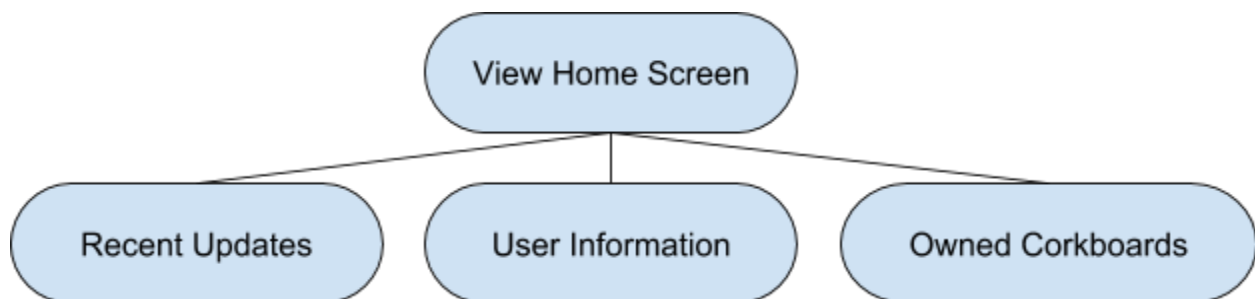
Login



Abstract Code

- User enters *e-mail address* and *PIN* into two input fields.
- If data validation for both fields is successful:
 - When **Login** button is clicked:
 - `-- Login User`
`SELECT email FROM "User"`
`WHERE email='$email' AND pin='$pin';`
 - If User is found, but password is incorrect:
 - Return to the Login Form with an error message.
 - If User is not found:
 - Return to the Login Form with an error message.
 - Else:
 - Continue to the Home Page.

View Home Screen



Abstract Code

- User logs in successfully or clicks the application's **logo** to view the **Home Page**.
- Run the **View Home Screen** task: query for the User's information utilizing the current ID of the User logged into the system from the HTTP session.
 - Find the first and last name of the current User and display them on the UI.
 - `-- Get User Information`

```

SELECT CONCAT("User".first_name, ' ', "User".last_name)
FROM "User"
WHERE email='$email';

```

- Find the Corkboards that the current User **owns**, **watches**, or is **owned** by a User that the currently logged in User is **following**.

- **-- Get Owned Corkboards**

```

SELECT "Corkboard".id AS homescreen_id,
       CONCAT("User".first_name, ' ',
              "User".last_name) AS homescreen_owner_name,
       "Corkboard".title AS homescreen_title,
       "Corkboard".is_private AS homescreen_is_private,
       "Pushpin".time_added AS homescreen_time_added
INTO OwnedCorkboards
FROM "Corkboard"
INNER JOIN "User"
    ON "User".email = '$email'
INNER JOIN "Pushpin"
    ON "Pushpin".corkboard_id = "Corkboard".id
WHERE "Corkboard".owner = '$email';

```

- **-- Get Watched Corkboards**

```

SELECT "Corkboard".id AS homescreen_id,
       CONCAT("User".first_name, ' ',
              "User".last_name) AS homescreen_owner_name,
       "Corkboard".title AS homescreen_title,
       "Corkboard".is_private AS homescreen_is_private,
       "Pushpin".time_added AS homescreen_time_added
INTO WatchedCorkboards
FROM "Corkboard"
INNER JOIN "Watched"
    ON "Watched".corkboard_id = "Corkboard".id
    AND "Watched".user_email = '$email'
INNER JOIN "User"
    ON "User".email = '$email'
INNER JOIN "Pushpin"
    ON "Pushpin".corkboard_id = "Corkboard".id;

```

- **-- Get Followed Corkboards**

```

SELECT "Corkboard".id AS homescreen_id,
       CONCAT("User".first_name, ' ',
              "User".last_name) AS homescreen_owner_name,
       "Corkboard".title AS homescreen_title,
       "Corkboard".is_private AS homescreen_is_private,

```

```

        "Pushpin".time_added AS homescreen_time_added
    INTO FollowedCorkboards
    FROM "Corkboard"
    INNER JOIN "User"
        ON "User".email = '$email'
    INNER JOIN "Pushpin"
        ON "Pushpin".corkboard_id = "Corkboard".id
    WHERE "Corkboard".owner IN
        (SELECT "Followed".followed_user_email
         FROM "Followed"
         WHERE "Followed".user_email = '$email');

```

- For each Corkboard that meets the above criteria:
 - Find the time that each Pushpin was **placed on** each Corkboard and sort the Pushpins by most recent update.
 - Sort each Corkboard by most recent Pushpin update.
 - Display the four most recently updated Corkboards.

- -- Display the four most recently updated Corkboards

```

CREATE TABLE "HomeScreenCorkboards" AS
    SELECT *
    FROM(
        SELECT *
        FROM OwnedCorkboards
        UNION ALL
        SELECT *
        FROM WatchedCorkboards
        UNION ALL
        SELECT *
        FROM FollowedCorkboards
    ) combined_tables;

```

```

SELECT
    "HomeScreenCorkboards".homescreen_time_added,
    "HomeScreenCorkboards".homescreen_id,
    "HomeScreenCorkboards".homescreen_title,
    "HomeScreenCorkboards".homescreen_owner_name,
    "HomeScreenCorkboards".homescreen_is_private,
FROM "HomeScreenCorkboards"
GROUP BY
    "HomeScreenCorkboards".homescreen_time_added,
    "HomeScreenCorkboards".homescreen_id,
    "HomeScreenCorkboards".homescreen_title,
    "HomeScreenCorkboards".homescreen_owner_name,
    "HomeScreenCorkboards".homescreen_is_private

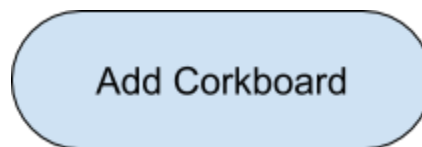
```

```
ORDER BY "HomeScreenCorkboards".homescreen_time_added
DESC
LIMIT 4;
```

- Find the Corkboards that the current User **owns**.
 - Display each of the **owned** Corkboards sorted by *title*, in alphabetical order.

```
-- Get My Corkboards
SELECT "Corkboard".id,
       "Corkboard".title,
       "Corkboard".is_private,
       COUNT(*) AS pushpin_count
FROM "Corkboard"
INNER JOIN "Pushpin"
  ON "Pushpin".corkboard_id = "Corkboard".id
WHERE "Corkboard".owner = '$email'
GROUP BY "Corkboard".title,
         "Corkboard".id,
         "Corkboard".is_private,
         pushpin_count
ORDER BY "Corkboard".title ASC;
```

Add Corkboard

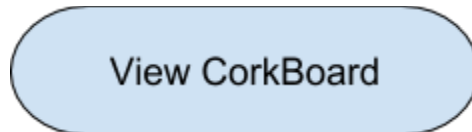


Abstract Code

- User clicks the **Add CorkBoard** button
- **Add Corkboard Form** will pop up, and User needs to:
 - Enter *title*
 - Select *category* from: Education, People, Sports, Other, Architecture, Travel, Pets, Food & Drink, Home & Garden, Photography, Technology, Art
 - Select *visibility*.
 - If *private*:
 - User must enter a *password*.
 - check the password
- User clicks the **Add** button.
 - Form data is validated.
 - New CorkBoard should be created with the current User set as its **owner**.
 - -- Add Corkboard

```
INSERT INTO "Corkboard" (title, is_private, password,
owner, category)
VALUES ('$title', '$is_private', '$password', '$owner',
'$category');
```

View Corkboard



Abstract Code

- After **User** runs the **View Home Screen** task, Run the **View CorkBoard** task: query for more recent CorkBoard updates as well as the user's own CorkBoard(s) utilizing the current ID of the user logged into the system from the HTTP session.
 - Run the **View CorkBoard** task: query for most recent CorBoards and the user's own CorkBoard(s) utilizing the current ID of the user logged into the system from the HTTP session.
 - Show a thumbnail of each PushPin image in the middle.
 - Hyperlink the thumbnail image to View Pushin page.
 - Show the owner's name, the title, when it was last updated and the category on the top, and the number of users currently watching the board on the bottom.
 - Find the first and last name of the current **User** and display them on the UI.
 - If the CorkBoard does not belong to the user, display the "follow" button on the top.
 - If the CorkBoard does not belong to the user, and the CorkBoard is public, the "watch" button will appear somewhere on the page.
 - If the current user clicks "watch" then the CorkBoard is added to the current user's watch list.
 - If clicking the "follow" button, the CorkBoard's owner is added to the current user's follow list.
 - If the CorkBoard belongs to the user, display the "add PushPin" button.
 - If clicking this button shows the **Add PushPin screen**, run **View Pushpin Task**
 - **-- View Corkboard**

```
SELECT "Corkboard".title,
      "Corkboard".category,
      CONCAT("User".first_name, ' ', "User".last_name)
      (SELECT "Pushpin".time_added
```

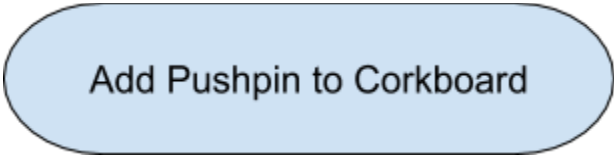
```

        FROM "Pushpin"
        WHERE "Pushpin".corkboard_id = '$corkboard_id'
        GROUP BY "Pushpin".time_added
        ORDER BY "Pushpin".time_added
        LIMIT 1) AS last_update
FROM "Corkboard"
INNER JOIN "User"
    ON "User".email = "Corkboard".user_email
WHERE "Corkboard".id='$corkboard_id';

--Get Watcher Count
SELECT COUNT(*)
FROM "Watched"
WHERE "Watched".corkboard_id = '$corkboard_id';

```

Add Pushpin to Corkboard



Add Pushpin to Corkboard

Abstract Code

- User selects **Add Pushpin** button and starts the **Add Pushpin to Corkboard** task.
 - User enters the *URL* of the image, a *description*, and multiple *tags*.
 - Validate that the description is less than 200 characters.
 - Validate the URL string and image file format.
 - parse the string, extract each of the tags and store them individually in the database
 - An individual tag should be 20 characters or less.
 - multiple tags are separated by commas.
 - Tags themselves cannot contain commas but may have any other character.
 - -- Add Tags from Pushpin entry form (note: \$tags is a comma delimited entry)


```

INSERT INTO "Tag" (pushpin_id, tag_text)
VALUES ('$pushpin_id', REGEXP_SPLIT_TO_TABLE('$tags',
E', '));

```
 - If form is valid:
 - Create the Pushpin, which is on the Corkboard.
 - -- Add Pushpin


```

INSERT INTO "Pushpin" (description, image_link, time_added,

```

```
corkboard_id)
VALUES ('$description', '$image_link', CURRENT_TIMESTAMP,
'$corkboard_id');
```

- Else:
 - Display a validation error on the form.

Follow/Unfollow User



Follow/Unfollow User


Abstract Code

- User selects the **Follow User** button and starts the **Follow User** task.
 - If User is not already **following** the other User:
 - A relationship is created for the currently logged in User and the selected User.
 - The **Follow User** button becomes “**Unfollow User**”.
 - `-- Follow User`

```
INSERT INTO "Followed" (user_email, followed_user_email)
VALUES ('$user_email', '$followed_user_email');
```
 - Else:
 - Delete the existing relationship between the two Users.
 - `-- Unfollow User`

```
DELETE FROM "Followed"
WHERE user_email='$user_email'
AND followed_user_email = '$followed_user_email';
```

Watch/Unwatch Corkboard



Watch/Unwatch Corkboard

Abstract Code

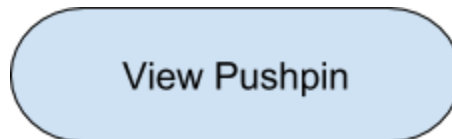
- User selects the **Watch Corkboard** button and starts the **Watch Corkboard** task.
 - If User is not already **watching** the Corkboard:

- A relationship is created for the currently logged in User and the selected Corkboard.
- The **Watch Corkboard** button becomes “**Unwatch Corkboard**”.
- **-- Watch a Corkboard**

```
INSERT INTO "Watched" (user_email, corkboard_id)
VALUES ('$user_email', '$corkboard_id');
```
- Else:
 - Delete the existing relationship between the User and the Corkboard.
 - **-- Unwatch a Corkboard**

```
DELETE FROM "Watched"
WHERE user_email='$user_email' AND corkboard_id
='$corkboard_id';
```

View Pushpin



Abstract Code

- User selects Pushpin from Corkboard
- -OR-
- User selects Pushpin from search results
- **View Pushpin** task runs
 - **View Pushpin** screen displays, and presents the following data:
 - Owner of its parent Corkboard.
 - Date and Time Pushpin was added.
 - Parent Corkboard's Title.
 - The image itself
 - Website address image comes from.
 - Description
 - List of Tags
 - **--Get Pushpin's Tags**

```
SELECT "Tag".tag_text
FROM "Tag"
WHERE "Tag".pushpin_id = '$pushpin_id';
```
 - Comments: The list of comments is sorted by the date and time that the comment was added
 - For each comment, list the name of the user who made the comment and the text of the comment (required), date and time (optional).
 - **--Get Pushpin Comments**

```

SELECT CONCAT("User".first_name, ' ',
              "User".last_name),
       "Comment".comment_text,
       "Comment".time_added
FROM "Comment"
INNER JOIN "Commented"
  ON "Commented".comment_id = "Comment".id
INNER JOIN "User"
  ON "User".email = "Commented".user_email
WHERE "Commented".pushpin_id = '$pushpin_id'
GROUP BY "Comment".time_added,
         "Comment".comment_text,
         "User".last_name,
         "User".first_name
ORDER BY "Comment".time_added DESC;

```

- A text box for user to comment.
- Liked by whom.

- --Get Users that Liked Pushpin

```

SELECT CONCAT("User".first_name, ' ',
              "User".last_name)
FROM "User"
INNER JOIN "Liked"
  ON "Liked".user_email = "User".email
WHERE "Liked".pushpin_id = '$pushpin_id';

```

- -- View Pushpin

```

SELECT
  "Pushpin".description,
  "Pushpin".image_link,
  SUBSTRING("Pushpin".image_link
            FROM '(:.*://)?(:www\.)?([^\/*]')
  "Pushpin".time_added,
  "Corkboard".title,
  CONCAT("User".first_name, ' ', "User".last_name)
FROM "Pushpin"
INNER JOIN "Corkboard"
  ON "Corkboard".id = "Pushpin".corkboard_id
INNER JOIN "User"
  ON "User".email = "Corkboard".owner
WHERE "Pushpin".id = '$pushpin_id';


```

- Screen allows Users to Like Pushpin

- If Liked:
 - Run **Like/Unlike Pushpin** task.
 - User can only Like a Pushpin once.

- User cannot like his or her own PushPins.
- Screen allows Users to Unlike a Pushpin that they previously Liked
 - If Unliked:
 - Run **Like/Unlike Pushpin** task.
 - User can Like the Pushpin again.
- Screen allows User to Comment on Pushpin.
 - If User comments:
 - Run **Comment on Pushpin** task.
 - User can comment multiple times.
 - User can also comment on his or her own PushPin.
 - Comments cannot be deleted.

Comment on Pushpin




Comment on Pushpin

Abstract Code

- User selects the **Comment** button, and starts the **Comment on Pushpin** task.
 - User enters the desired *text* in an input box.
 - User selects the **Submit** button.
 - If *text* is not blank:
 - Comment *text* is applied to the Pushpin identified by the current time.
 - --Add Comment

```
WITH comment_insert AS (INSERT INTO "Comment"
(comment_text, time_added)
VALUES ('$Comment_Text', CURRENT_TIMESTAMP)
RETURNING "Comment".id AS insert_comment_id)
INSERT INTO "Commented" (comment_id, user_email,
pushpin_id)
SELECT insert_comment_id, '$user_email',
'$pushpin_id' FROM comment_insert;
```
 - Comment is displayed on the UI.
 - Else:
 - Display a validation error on the UI.

Like/Unlike Pushpin


 A light blue rounded rectangular button with a thin black border, containing the text "Like/Unlike Pushpin" in a dark grey sans-serif font.

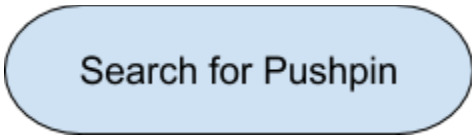
Abstract Code

- User selects the **Like** button, and starts the **Like/Unlike Pushpin** task.
 - Relation is created between the User ID and the Pushpin ID, representing that the current User likes the currently viewed Pushpin.
 - The relationship is shown on the UI after the transaction is complete.
 - **-- Like a Pushpin**

```
INSERT INTO "Liked" (user_email, pushpin_id)
VALUES ('$user_email', '$pushpin_id');
```
 - **Like** button becomes **Unlike** button
- User selects the **Unlike** button, and starts the **Like/Unlike Pushpin** task.
 - Relation is removed.
 - **-- Unlike a Pushpin**

```
DELETE FROM "Liked"
WHERE user_email='$user_email' AND pushpin_id ='$pushpin_id';
```

Search for Pushpin


 A light blue rounded rectangular button with a thin black border, containing the text "Search for Pushpin" in a dark grey sans-serif font.

Abstract Code

- User enter a *search term* in the search textbox.
- -OR-
- User selects a **Popular Tag**, which is then the *search term*.
- **Search for Pushpin** task runs:
 - If the *search term* is contained in either the:
 - Description,
 - **-- Search Pushpins (description)**

```
SELECT "Pushpin".description,
      "Corkboard".title,
      CONCAT("User".first_name, ' ',
            "User".last_name)
```

```

FROM "Pushpin"
INNER JOIN "Corkboard"
    ON "Corkboard".id = "Pushpin".corkboard_id
INNER JOIN "User"
    ON "User".email = "Corkboard".owner
WHERE "Pushpin".description LIKE
    '%$description_search_input%'
    AND "Pushpin".is_private = '0'
GROUP BY "Pushpin".description
ORDER BY "Pushpin".description ASC;

```

■ One of the Pushpin's Tags, or

- --Search Pushpins (tag)

```

SELECT "Pushpin".description,
    "Corkboard".title,
    CONCAT("User".first_name, ' ',
        "User".last_name)
FROM "Tag"
INNER JOIN "Pushpin"
    ON "Pushpin".id = "Tag".pushpin_id
INNER JOIN "Corkboard"
    ON "Corkboard".id = "Pushpin".corkboard_id
INNER JOIN "User"
    ON "User".email = "Corkboard".owner
WHERE "Tag".tag_text LIKE '%$tag_search_input%'
    AND "Pushpin".is_private = '0'
GROUP BY "Pushpin".description
ORDER BY "Pushpin".description ASC;

```

■ The Category of the Pushpin's parent Corkboard

- -- Search Pushpins (category)

```

SELECT "Pushpin".description,
    "Corkboard".title,
    CONCAT("User".first_name, ' ',
        "User".last_name)
FROM "Corkboard"
INNER JOIN "User"
    ON "User".email = "Corkboard".owner
INNER JOIN "Pushpin"
    ON "Pushpin".corkboard_id = "Corkboard".id
WHERE "Corkboard".category LIKE
    '%$category_search_input%'
    AND "Pushpin".is_private = '0'
GROUP BY "Pushpin".description
ORDER BY "Pushpin".description ASC;

```

- Then display the following information from the Pushpin:
 - Description (which is also a link to **View Pushpin** screen)
 - Parent Corkboard
 - Corkboard Owner
- The results are sorted alphabetically by Description
- Private Pushpins never show in results.

View Popular Tags

View Popular Tags

Abstract Code

- User selects the **Popular Tags** button.
- **View Popular Tags** task runs.
 - **View Popular Tags** screen displays
 - List the top five tags with the most PushPins.
 -
 - Fetch the most used unique Tags on all Pushpins.
 - If Pushpin has the same Tag twice:
 - Only count one instance.
 - Order Tags descending by Pushpin count:
 - **--Get Popular Tags and Counts**

```
SELECT "Tag".tag_text, COUNT(*) AS tag_count
FROM "Tag"
GROUP BY 1
ORDER BY tag_count DESC
LIMIT 5;
```
 - If a Tag name is clicked:
 - Perform **Search For Pushpin** task.

View Popular Sites

View Popular Sites

Abstract Code

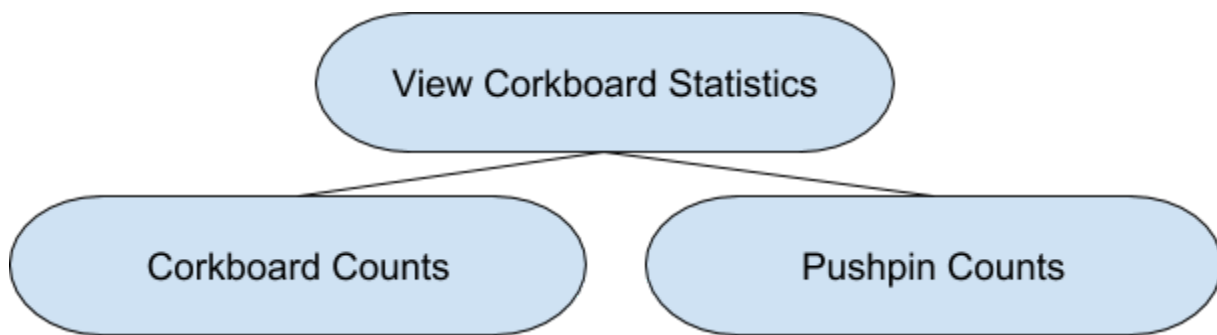
- Run the **View Popular Sites** task.

[Table of Contents](#)

- Fetch the most common URLs (and their counts) that appear on Pushpins.
- **--Get Popular Sites and Counts**

```
SELECT SUBSTRING("Pushpin".image_link
                FROM '(:.*://)?(?:www\.)?([^\/*]*)'),
       COUNT(*) AS site_count
FROM "Pushpin"
GROUP BY 1
ORDER BY site_count DESC;
```
- Display the information on the UI.

View Corkboard Statistics



Abstract Code

- Run the **View Corkboard Statistics** task.
 - For each User in the system:
 - Display the total number of public Corkboards that each User **owns**.
 - Display the total number of private Corkboards that each User **owns**.
 - **--Get Count of User's Public and Private Corkboards**

```
SELECT
    SUM(CASE WHEN "Corkboard".is_private = '0' THEN 1
            ELSE 0 END) AS public_corkboards,
    SUM(CASE WHEN "Corkboard".is_private = '1' THEN 1
            ELSE 0 END) AS private_corkboards
FROM "Corkboard"
WHERE "Corkboard".owner = '$user_email';
```
 - Display the total number of Pushpins that are on public Corkboards **owned** by each User.
 - Display the total number of Pushpins that are on private Corkboards **owned** by each User.
 - **--Get Count of Pushpins on User's Public and Private Corkboards**

```
SELECT
    SUM(CASE WHEN "Corkboard".is_private = '0' THEN 1
```

```
        ELSE 0 END) AS public_pushpins,  
    SUM(CASE WHEN "Corkboard".is_private = '1' THEN 1  
        ELSE 0 END) AS private_pushpins  
FROM "Corkboard"  
INNER JOIN "Pushpin"  
    ON "Pushpin".corkboard_id = "Corkboard".id  
WHERE "Corkboard".owner = '$user_email';
```