Product Owner for Sprint 1 - Suyeon

Product Backlog

1. User Authentication & Spotify Integration

- **Login using Spotify Authorization**: Users must connect their Spotify account via OAuth authentication to load user's frequently listened to songs. (https://developer.spotify.com/)
    - After login through spotify, users can create unique usernames

2. Playlist Creation & Collaboration

- **Home Page - Create Shared Playlist**: Users can initiate a shared playlist session.
- **Customize Playlist Length**: Users can define the length of the playlist in minutes.
- **Invite Friends**: A shareable link that allows friends to join and participate in the playlist creation process.
    - Friends must also connect their Spotify accounts to participate.
- **Adding Songs:** Participants of the shared playlist will be presented with their unique list of recently played songs that they can choose to add to the shared playlist.
- **Start Playlist Creation:** A button to initiate the voting stage.

3. Voting Mechanism

- Each participant has **1 minute** to vote for or against each song.
- A short audio preview of the song can be available for users who need to listen to the song. (see if feasible)
- Participants can vote by swiping left or right.

4. Playlist Generation & Sharing

- **Final Playlist Compilation**: Once voting ends, the system generates a Spotify playlist based on highly voted songs. This will only appear for the playlist creator. Others are automatically added as collaborators.
- **Playlist Naming**: The playlist creator must provide a name for the finalized playlist.
- **Social Sharing**:
    - The playlist can be posted to a general feed.
    - The generated playlist is also saved to the creator's profile for future access.

5. Playlist Stream

- **Like and Dislike**: The stream will display all playlists shared by your friends, allowing you to like or dislike them.

Data Models

```
class User(AbstractUser):
    """
    Extended User model for storing Spotify-related information
    """
    spotify_token = models.CharField(max_length=255, blank=True, null=True)
    friends = models.ManyToManyField('self', symmetrical=True, blank=True)
    liked_playlists = models.ManyToManyField('Playlist', related_name='liked_by', blank=True)

    def __str__(self):
        return self.username


class Song(models.Model):
    """
    Model to store song information
    """
    spotify_id = models.CharField(max_length=100, unique=True)
    title = models.CharField(max_length=255)
    artist = models.CharField(max_length=255)
    album = models.CharField(max_length=255, blank=True, null=True)
    duration_ms = models.IntegerField(default=0)

    def __str__(self):
        return f"{self.title} - {self.artist}"
```

```python
class Playlist(models.Model):
    """
    Model for completed playlists
    """
    id = models.UUIDField(primary_key=True, default=uuid.uuid4, editable=False)
    name = models.CharField(max_length=255)
    owner = models.ForeignKey(User, on_delete=models.CASCADE, related_name='created_playlists')
    collaborators = models.ManyToManyField(User, related_name='collaborated_playlists', blank=True)
    songs = models.ManyToManyField(Song, related_name='playlists', blank=True)
    created_at = models.DateTimeField(default=timezone.now)
    is_public = models.BooleanField(default=True)

    # Likes are tracked via the ManyToManyField in User (liked_playlists)

    def __str__(self):
        return self.name

    @property
    def likes_count(self):
        return self.liked_by.count()



class PlaylistDraft(models.Model):
    """
    Model for playlists under creation (before voting)
    """
    id = models.UUIDField(primary_key=True, default=uuid.uuid4, editable=False)
```

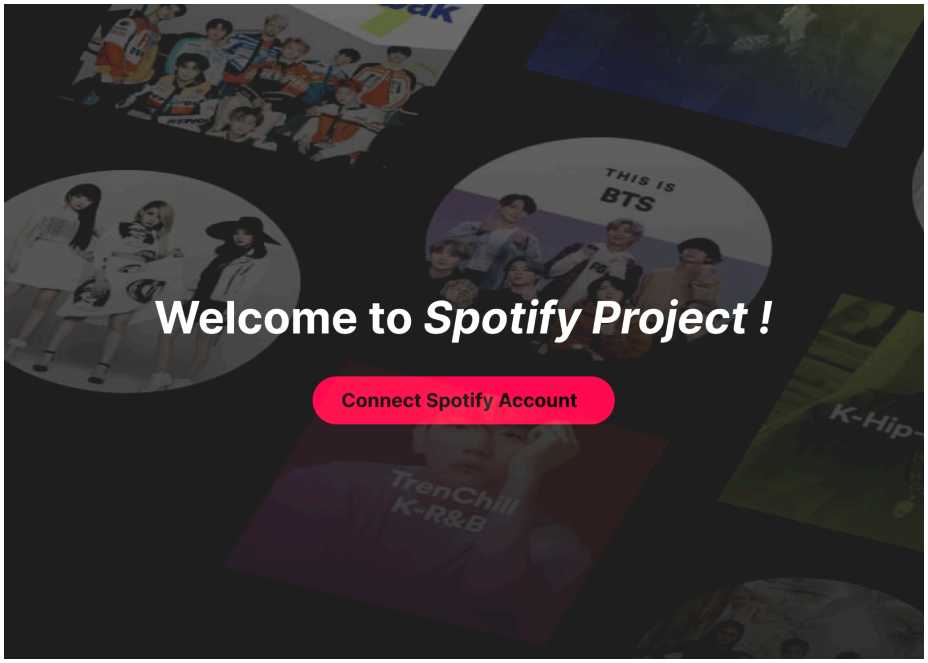
```
name = models.CharField(max_length=255)

owner = models.ForeignKey(User, on_delete=models.CASCADE, related_name='draft_playlists')

collaborators = models.ManyToManyField(User, related_name='collaborated_drafts', blank=True)

created_at = models.DateTimeField(default=timezone.now)

is_voting_complete = models.BooleanField(default=False)



def __str__(self):

    return f"Draft: {self.name}"
```

First Sprint Backlog

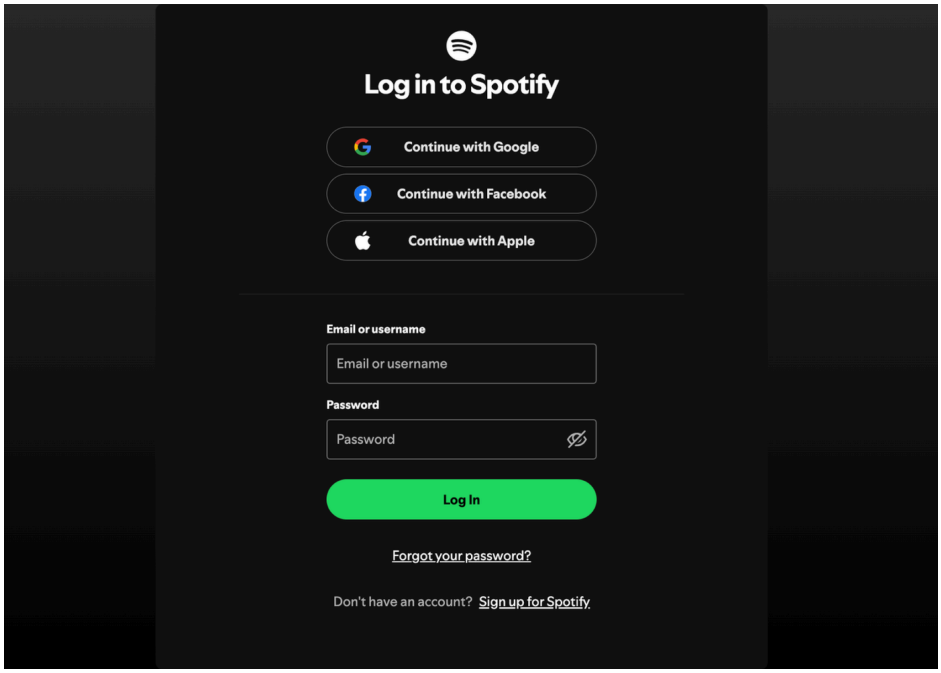
1. User login through Spotify (Vicky) + creating an account (Suyeon)
2. Integrating Spotify API
    a. retrieve the list of most listened songs (Vicky)
    b. create new playlist (Suyeon)
    c. add collaborators to the playlist (Suyeon)
3. Code HTML for UI of the home page (Suyeon will prototype, Vicky will figma)
    a. display existing playlists
    b. playlist creation page button

Wireframe

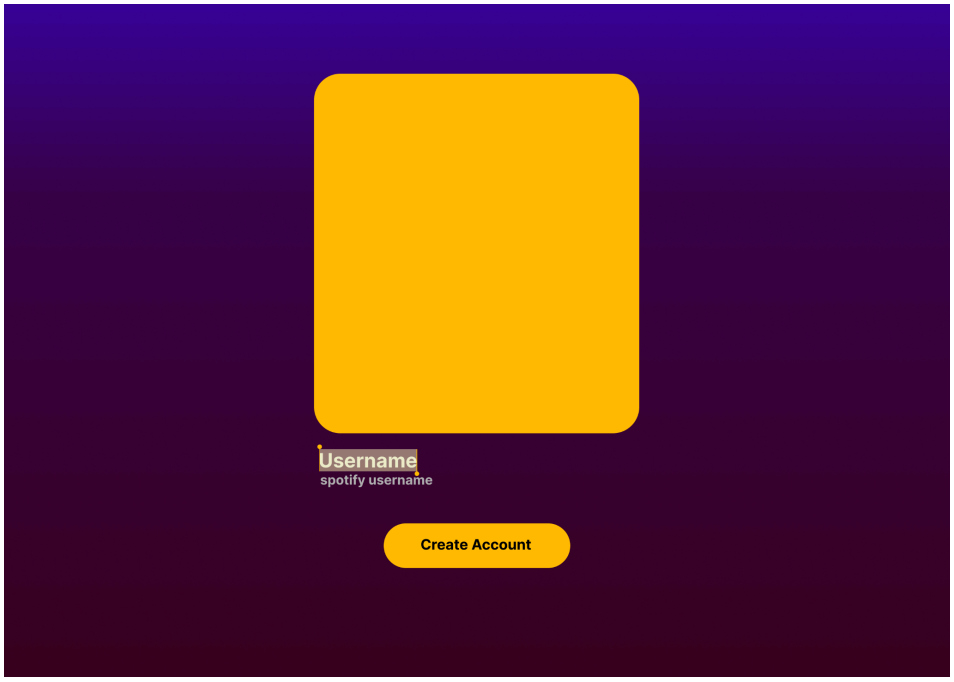
Login/Sign Up Page



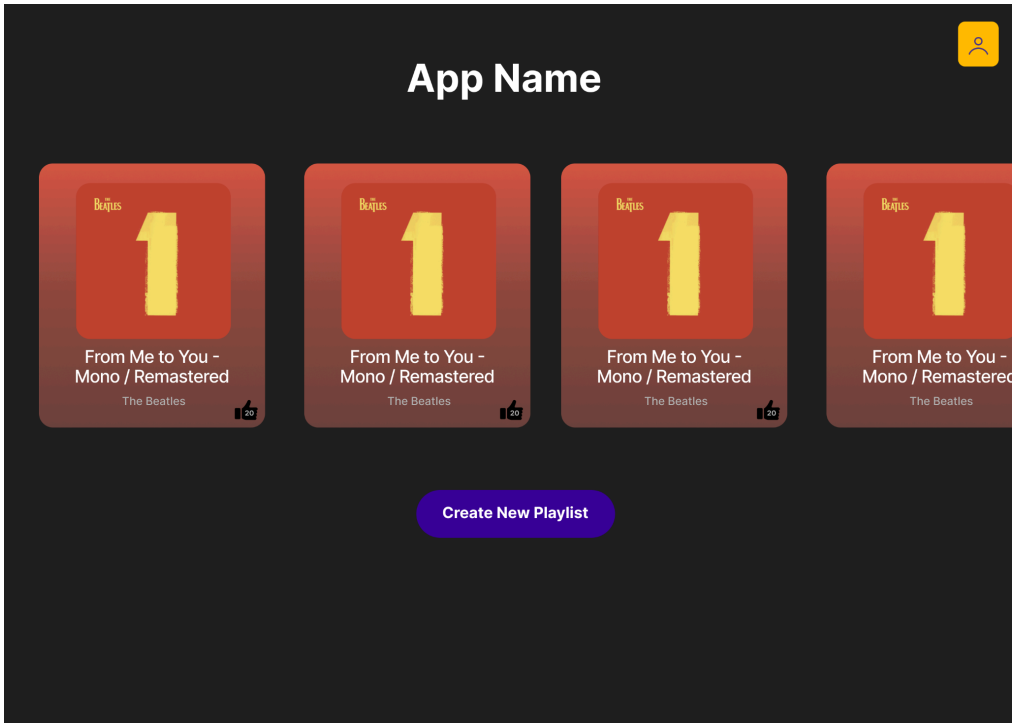
Spotify Login



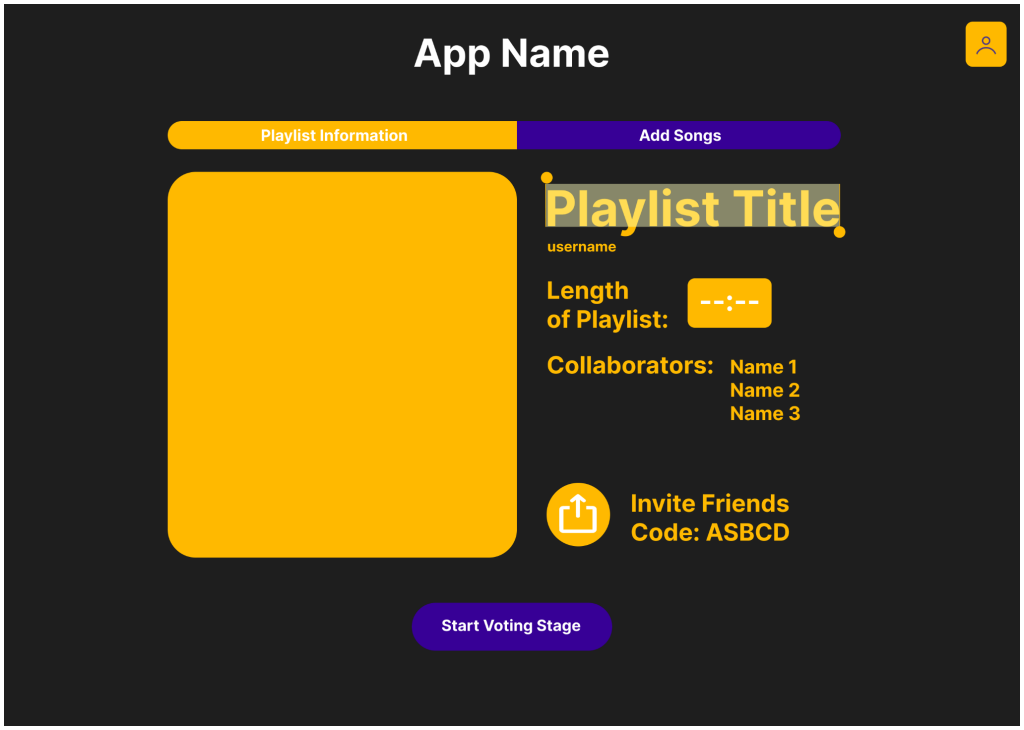
Username & Photo for Profile Creation



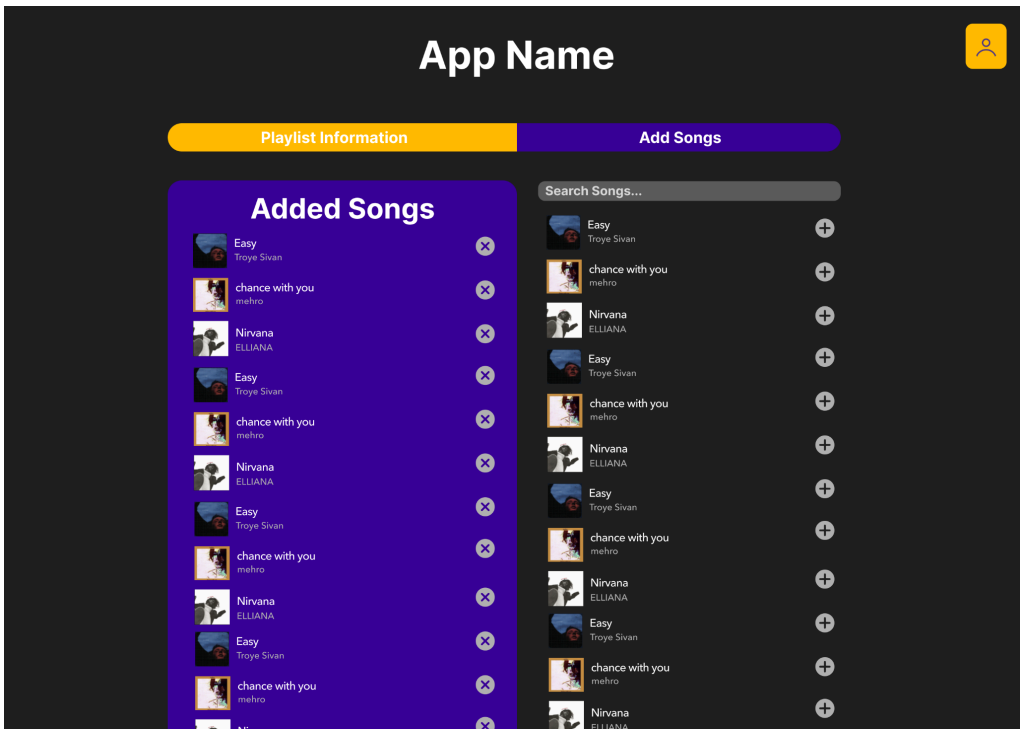
Home Page



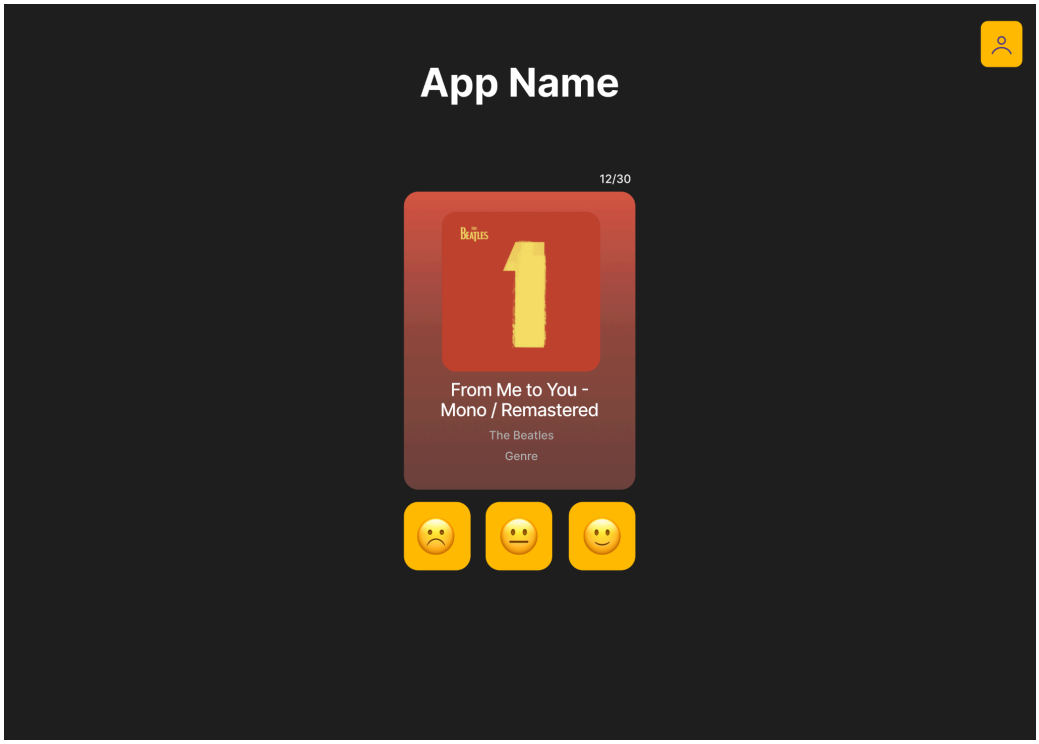
Create New Playlist Page



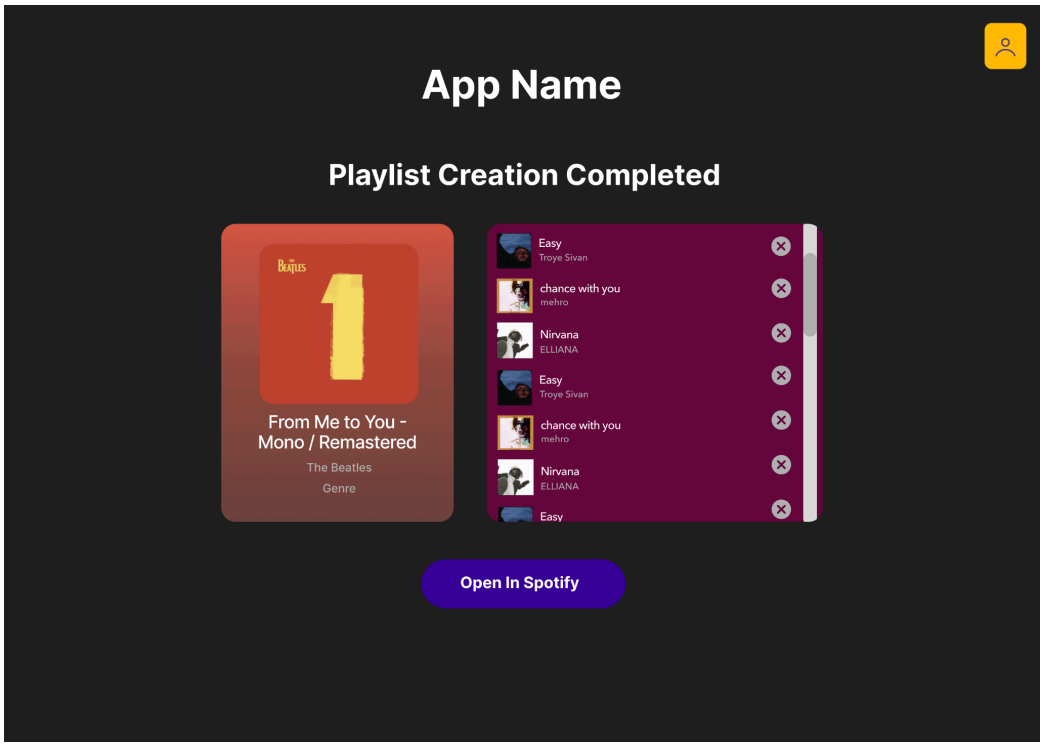
Add Songs Page



Voting Page



Playlist Complete Page

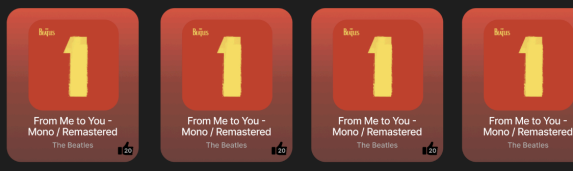


User Profile Page

App Name

Created Playlist

Liked Playlist



Username

spotify username