

POLITECNICO
MILANO 1863

Advance Coding Tools And Methodologies
29 june 2023

FACETUNE

Marcello Grati
Alessandra Moro
Silvia Pasin

- **Graphical User Interface**

Homepage



index.html



style.css

FACETUNE

CREATE YOUR AVATAR

AVATARS COMMUNITY

MARCELLO GRATI - ALESSANDRA MORO - SILVIA PASIN

Create your avatar



avatar_page.html



avatarpage_style.css



avatarpage_app.js

FACETUNE

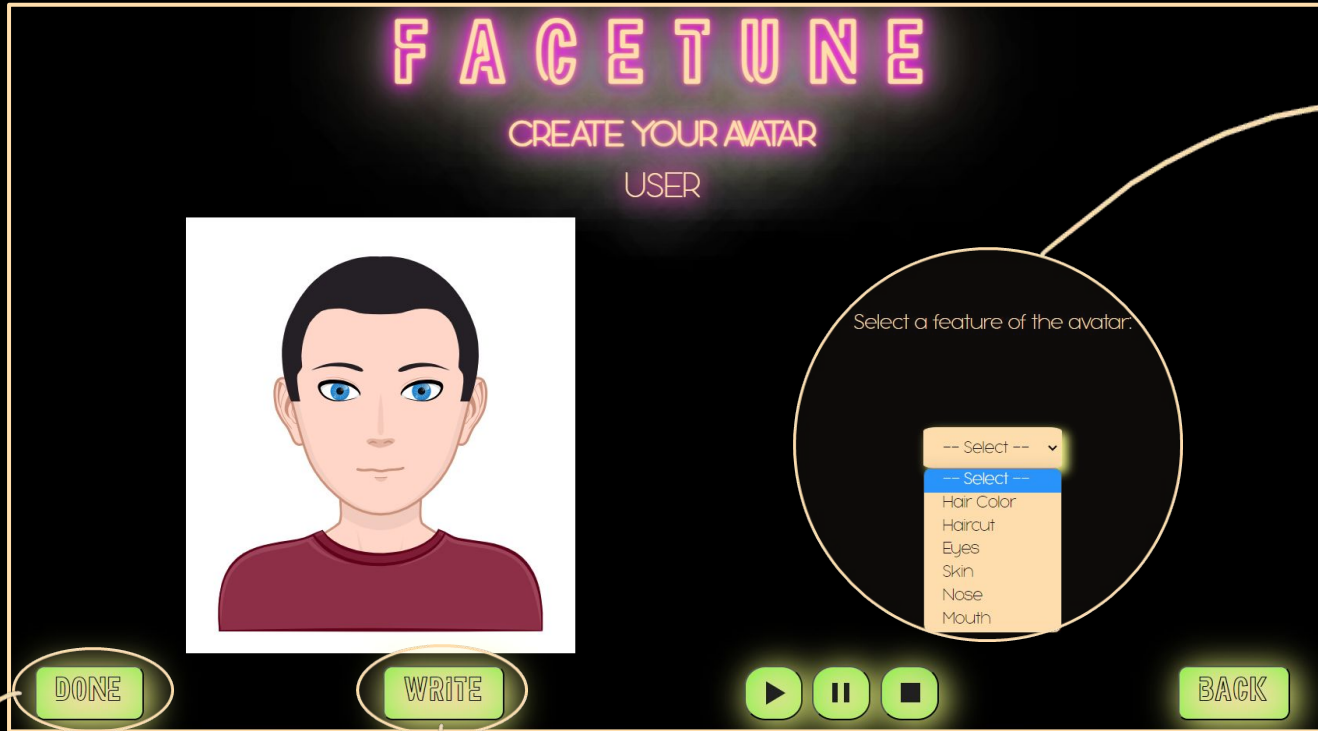
CREATE YOUR AVATAR

Nickname:

Confirm

BACK

Avatar creation:



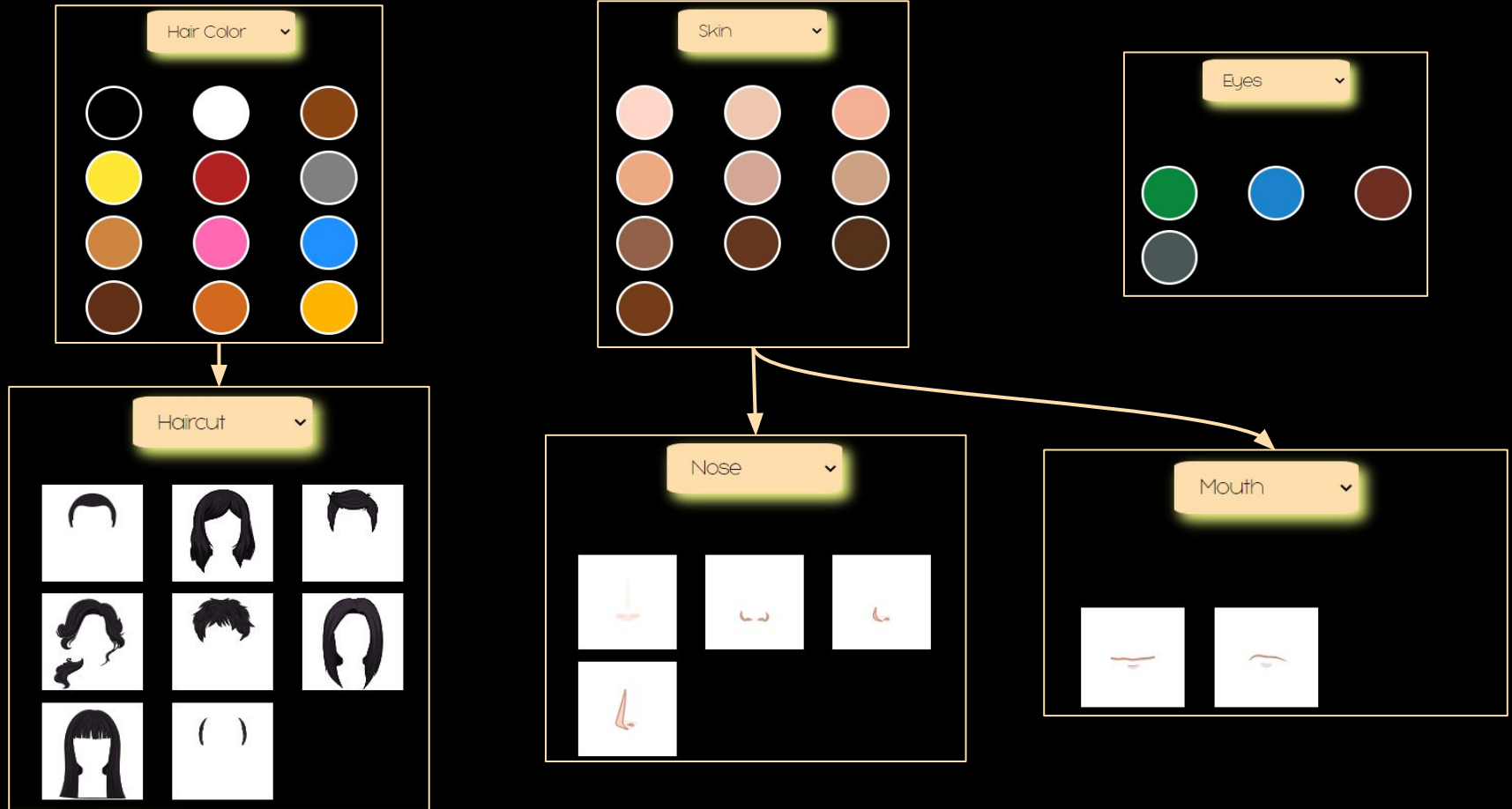
Selection of features:

- Hair color
- Haircut
- Eyes color
- Skin color
- Nose shape
- Mouth shape

Creation completed and
avatar saved in the
“Avatars community”

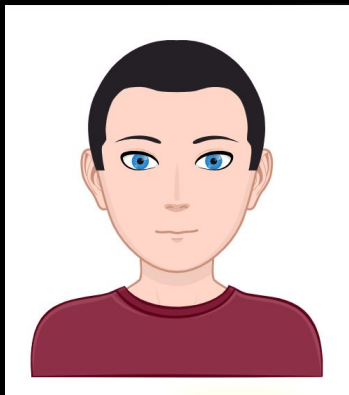
According to the chosen characteristics,
a tune is generated and played.
It’s still possible to modify the avatar.

Selection of features:





body.svg



The image of the avatar is created using Scalable Graphics Vector format. Each shape is created through the tag `<path>` and then, using the tag `<g>`, grouped together to form the various part of the body.

The elements whose color can be changed from the user, are assigned to a class defined in the .css according to the color chosen.

```
<g id="eyebrows_right" class="red hair">  
  <path id="SvgjsPath5354-5-0" d="[...]" stroke-width="0"  
    opacity="1"/>  
</g>
```

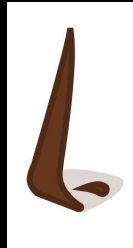


For what concerns haircut, nose and mouth, SVG images of the features are used inside of the main SVG (body.svg).

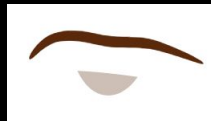
```
<use id="haircut" href="1h.svg#1h" class="brown_hair"/>
```



```
<use id="nose" href="type4n.svg#type4n" class="skin81"/>
```



```
<use id="mouth" href="type2m.svg#type2m" class="skin81"/>
```



FACETUNE

CREATE YOUR AVATAR

NICO



DOWNLOAD



BACK

Download of the tune created

Avatars Community



community_page.html



community_style.css



community_app.js

FACETUNE

AVATARS COMMUNITY

-- Filter by --

-- Filter by --

- Visualize all the avatars
- Hair Color
- Haircut
- Eyes
- Skin
- Nose
- Mouth
- Nickname

BACK

The screenshot shows the 'FACETUNE AVATARS COMMUNITY' app interface. At the top, the title 'FACETUNE' is in large, glowing pink letters, with 'AVATARS COMMUNITY' below it in smaller pink letters. A yellow search bar contains the text 'Filter by'. On the left, a yellow button labeled 'Order by' is circled in red. Below it, a list of names is shown: ALE, SILVIA, CLAUDIO, GABRI, and MATTIA. A vertical grey bar is positioned to the right of this list. In the center, a large avatar of a woman with blonde hair and green eyes is displayed, with the name 'ALE' above it. Below the avatar, the text 'SCORE: 4/5' is shown. On the right, a yellow button labeled 'VOTE' is circled in red, with five yellow stars below it. At the bottom, there are three yellow buttons: a play button, a pause button, and a stop button. A yellow 'BACK' button is located in the bottom right corner.



● Database management



app.js



facetune.db

Creation of table genoma:

```
const db = require('better-sqlite3')('facetune.db');  
db.prepare(  
  `CREATE TABLE IF NOT EXISTS genoma (  
    haircolor TEXT NOT NULL,  
    haircut TEXT NOT NULL,  
    eyes TEXT NOT NULL,  
    skin TEXT NOT NULL,  
    nose TEXT NOT NULL,  
    mouth TEXT NOT NULL,  
    username TEXT NOT NULL,  
    score REAL DEFAULT 0,  
    votes INTEGER DEFAULT 0,  
    id INTEGER DEFAULT 0  
  );`  
) .run();
```

Insertion of new avatar:

```
db.prepare(  
  `INSERT INTO genoma (haircolor, haircut, eyes, skin, nose, mouth, username, id)  
  VALUES (?, ?, ?, ?, ?, ?, ?, ?)`).run(...avatar_da_salvare);
```

Filtering of avatars:

```
const query = `SELECT * FROM genoma WHERE ${filters[0]} LIKE ?`;`;  
const filtered_avatars = db.prepare(query).all(filters[1]);
```

Example of 'filters': filters = [hair_color, red_hair]



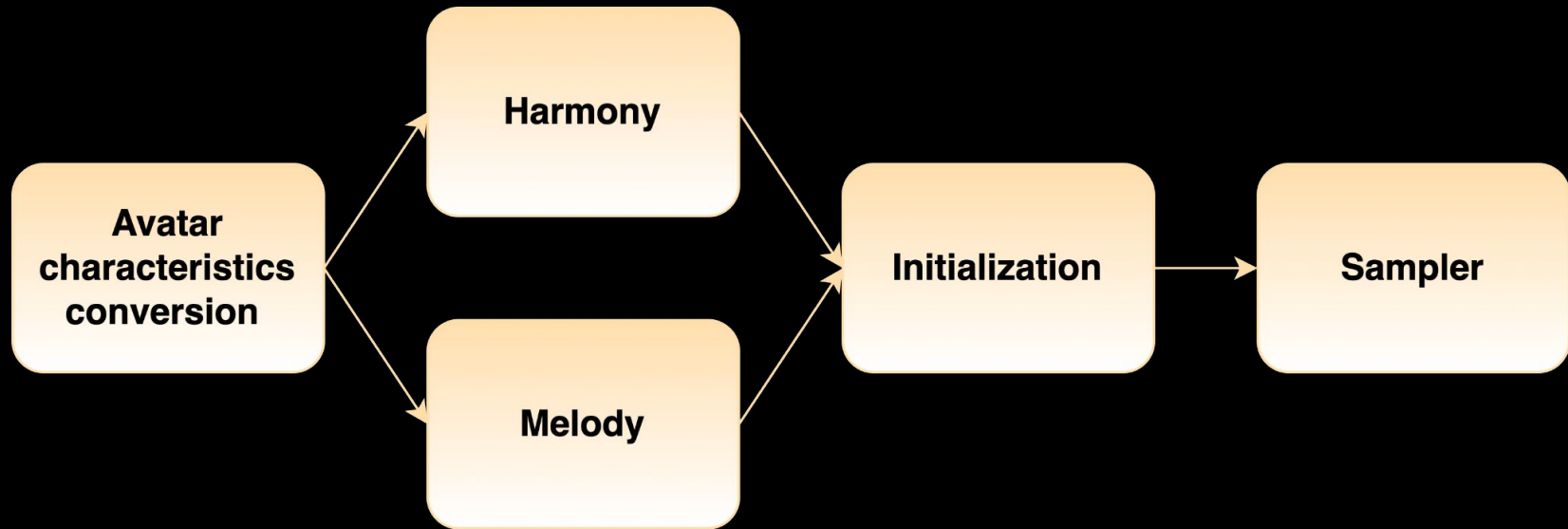
Updating the score:

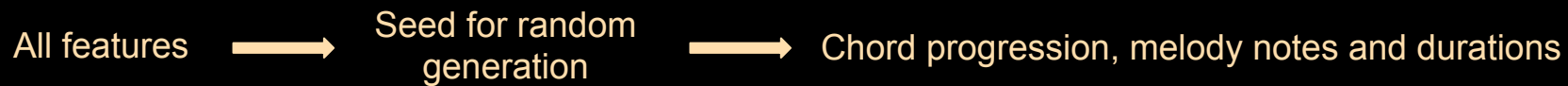
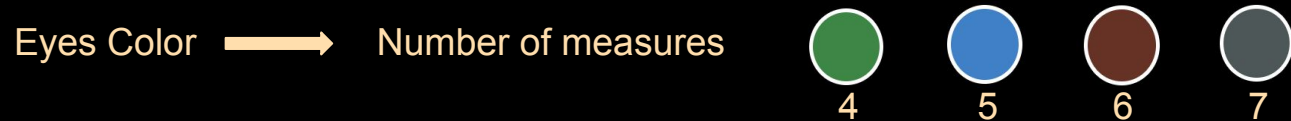
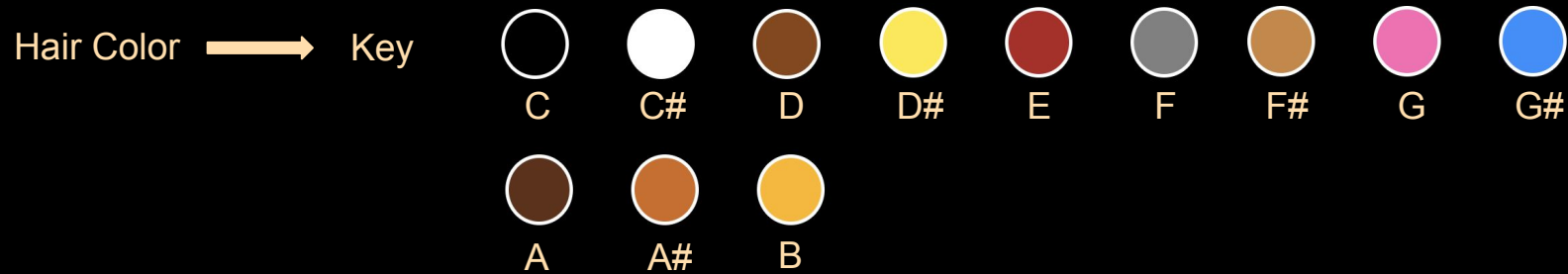
```
const existingAvatars = db.prepare('SELECT * FROM genoma WHERE id = ?').all(id);  
  
db.transaction(() => {  
  if (existingAvatars.length > 0) {  
    const updateStmt = db.prepare('UPDATE genoma SET score = ?, votes = ? WHERE id = ?');  
    existingAvatars.forEach(existingAvatar => {  
      updateStmt.run(score, votes, id);  
    });  
  }  
});
```

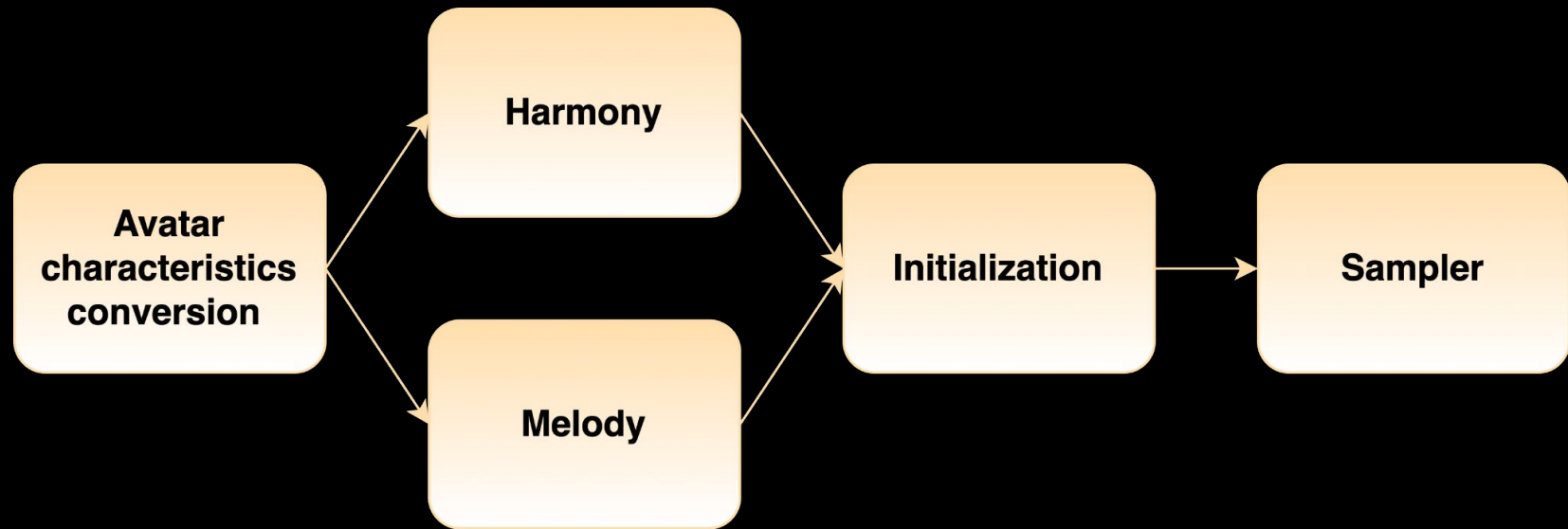
- **Music generation**



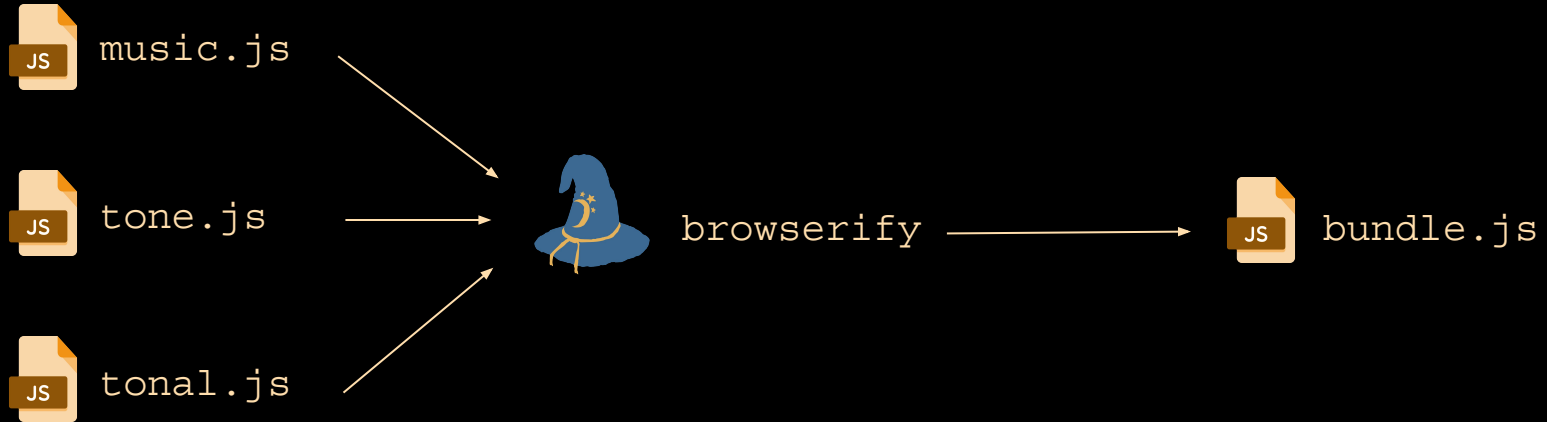
music.js

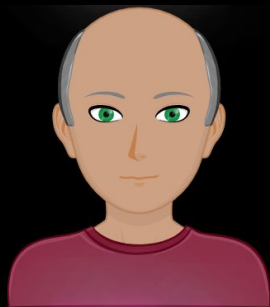
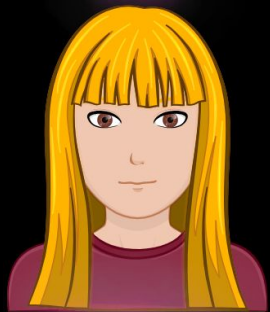






- Libraries





Thank you for the attention!

