EmojiPass Study

This document provides instructions for setting up and running the EmojiPass Study application for evaluation purposes. The application implements a comparative study platform examining emoji-based passwords versus traditional text-based passwords in terms of usability, memorability, and security.

> Try the live application at https://hcs-n0miya.vercel.app/

Table of Contents

Project Overview

EmojiPass investigates how emoji inclusion affects password:

- **Security**: Entropy, strength metrics, and resistance to observation attacks
- **Usability**: Creation time, recall speed, and user experience
- **Memorability**: Short-term and long-term recall success rates

Installation Prerequisites

- Node.js (v16 or higher)
- pnpm package manager (recommended for faster installation)

If pnpm is not installed, install it globally using npm:

```
npm install -g pnpm
```

Installation Instructions

1. Navigate to the project directory
2. Install dependencies:

```
pnpm install
```

Running the Application

Start the development server:

```
pnpm dev
```

The application will be available at http://localhost:5173 (or another port if 5173 is in use)

Application Structure & Usage Guide

1. **Landing Page**
   The landing page provides access to three experimental workflows:
   - Text Password Study
   - Emoji Password Study
   - Shoulder Surfing Experiment
2. **Text Password Path**
   - **Creation**: Generate a traditional text password (minimum 8 characters)
   - **Metrics**: View entropy, strength, and creation time metrics
   - **Short-term recall**: Test immediate memorability
   - **Long-term recall**: Return later to test delayed recall
3. **Emoji Password Path**
   - **Creation**: Generate a password using at least 4 emojis
   - **Metrics**: View comprehensive security metrics including emoji proportion
   - **Short-term recall**: Test immediate memorability
   - **Long-term recall**: Return later to test delayed recall
4. **Shoulder Surfing Experiment**
   - **Setup**: Select password type (emoji-mixed or text)
   - **Target mode**: One participant views and enters a randomly generated password
   - **Observer mode**: Another participant attempts to recreate the observed password

- **Results**: Analyze success rates and Levenshtein distance metrics

Key Implementation Features

1. **Secure Password Handling**
   - bcrypt-based password hashing with salt rounds
   - No plaintext password storage
   - Secure credential verification
2. **Emoji Processing Technology**
   - Cross-platform emoji rendering via Twemoji library
   - Grapheme-aware string handling for multi-codepoint emojis
   - Categorized emoji picker with 8 emoji groups
3. **Security Metrics Calculation**
   - Shannon entropy calculation for mixed character sets
   - Estimated crack-time based on current computational capabilities
   - Separate strength calculations for text and emoji components
4. **Memory Testing Framework**
   - Short-term recall testing (immediate memory)
   - Long-term recall testing (using localStorage persistence)
   - Success rate and attempt tracking
5. **Shoulder Surfing Security Testing**
   - Simulated observation attack scenarios
   - Levenshtein distance calculation for partial success measurement
   - Comparison of text vs. emoji resistance to observation

Data Collection

The application enables participants to copy results in JSON format for submission to a Microsoft Form. All experimental metrics are automatically calculated and formatted for easy collection.

Technical Implementation

This project is built with:

- React with TypeScript
- Tailwind CSS for styling
- bcryptjs for secure password handling
- React Router for navigation
- LocalStorage for session persistence

Project Structure

```
src/
├── components/         # Core UI components
│   ├── EmojiDisplay.tsx           # Emoji rendering
│   ├── EmojiPasswordInput.tsx     # Password input with emoji support
│   ├── EmojiPicker.tsx            # Categorized emoji selection
│   ├── PasswordMetrics.tsx        # Security metric display
│   └── PasswordStrengthMeter.tsx  # Visual strength indicator
├── pages/              # Application pages
│   ├── EmojiPasswordApp.tsx       # Emoji password flow
│   ├── HomePage.tsx               # Landing page
│   ├── ShoulderSurfingExperiment.tsx  # Security testing
│   └── TextPasswordApp.tsx        # Text password flow
├── utils/              # Utility functions
│   ├── emojiUtils.ts              # Emoji handling
│   ├── levenshteinUtils.ts        # Distance calculation
│   └── passwordUtils.ts           # Security algorithms
└── main.tsx           # Application entry point
```