CT30A3204 Advanced Web Applications Project Report

Junyang Huang, 000206529

03/03/2024

Contents

1. Introduction

The project is a dynamic social networking and chat application that redefines the way people connect online. I made an app that emphasizes simplicity and authenticity, allowing users to stand out by creating and customizing their own personal cards that showcase their interests, personality, and preferences.

The core appeal of the app is that users can browse through hand-picked personal cards and express their appreciation for others' profiles with a swipe. The interactive feature of "liking" another user's personal card opens up the possibility of conversation - mutual liking opens the door to private chats and creates a reciprocal environment where meaningful connections can blossom.

The goal of the app is to strip away the artificial layers that are common in virtual interactions and promote a direct, genuine approach to socializing. Through quick bilateral matches and chat features, the app offers users the opportunity to connect with like-minded individuals, encouraging users to make new friends or more in a friendly and welcoming online space.

2. Technology choices

Environment: Node.js

Front-end

Framework: React

Dependencies:

[

UI layout: @mui/material

UI icons: @mui/icons-material

Interactions: react-swipeable, react-spring, react-gesture

Navigations: react-router-dom

Others: moment

]

(For detailed dependencies, view package.json in the codes: ./client/package.json)

Back-end

Framework: Express

Dependencies:

[

Device: nodemon

Database: mongoose

Authentication: bcrypt, passport, jsonwebtoken, passport-jwt

Cross-domain communication: cors

Others: dotenv, multer

]

(For detailed dependencies, view package.json in the codes: ./server/package.json)

3. Installation guide & User manual

Make sure you have the Node.js environment installed on your computer.

Get the project's source code from the commit file or Github link. To save space, I have not uploaded node_modules and other third-party dependencies, so you will need to install them as a first step before running the application.

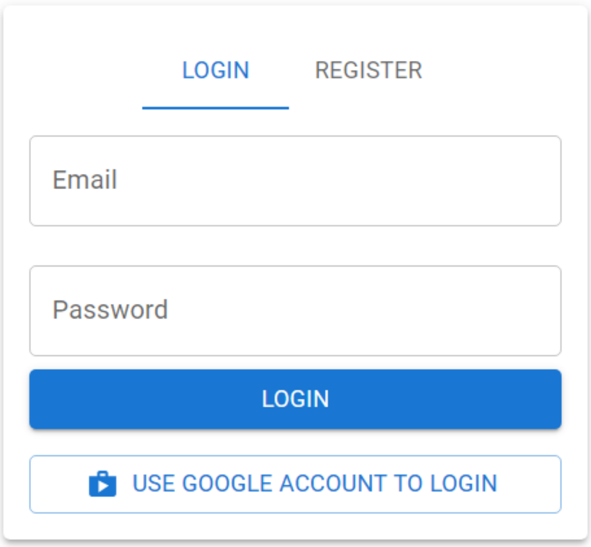
First, open a terminal in the root directory of the project and type the command `npm install` to install the project's basic dependencies. After that, enter command `cd client` and `cd server` to access the front-end and back-end folders of the project. In the frontend and backend folders, type the command `npm install` to install the dependencies for both the frontend and backend.

(You can also look for the package.json folder in the project root folder as well as the front and backend folders to determine which dependencies need to be installed).

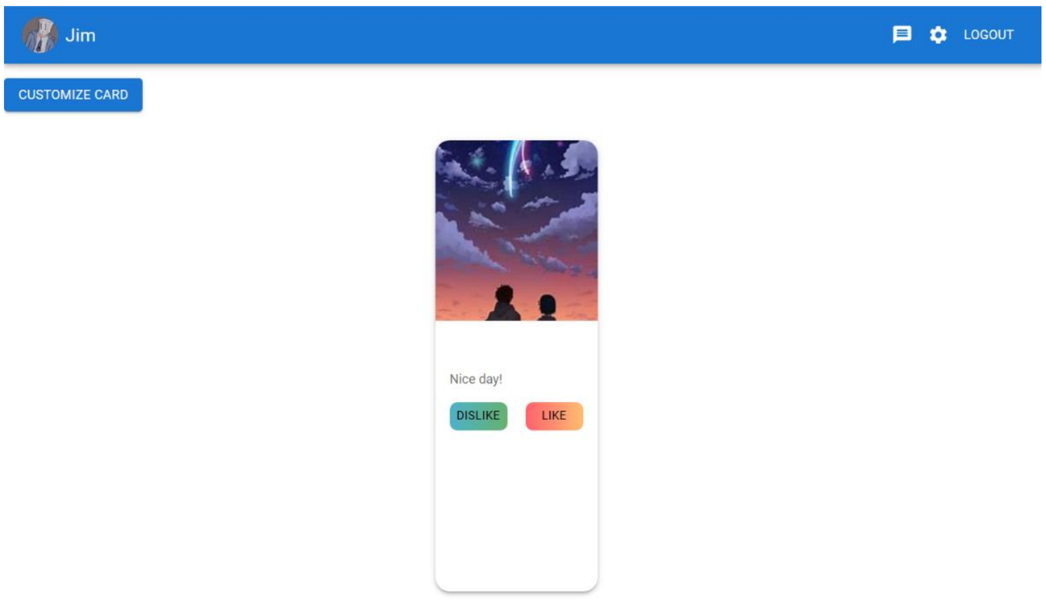
After all dependencies are installed, you should be able to run the application.

Open a new terminal in the project root directory (you can use PowerShell or VS Code), Type `cd server` to enter the backend folder, and `npm run dev` to start the server. Next, open a new terminal in the root directory of the project, type `cd client` to enter the front-end folder, and type `npm start` to start the client (web app). Once the application is compiled, it will open your browser and load the web page.

When you open the app for the first time, you will first see a login/registration page, you need to register a new account before you can start using the app, select the "Register" option, set your email, password, username and profile. Then you can login with your email and password.

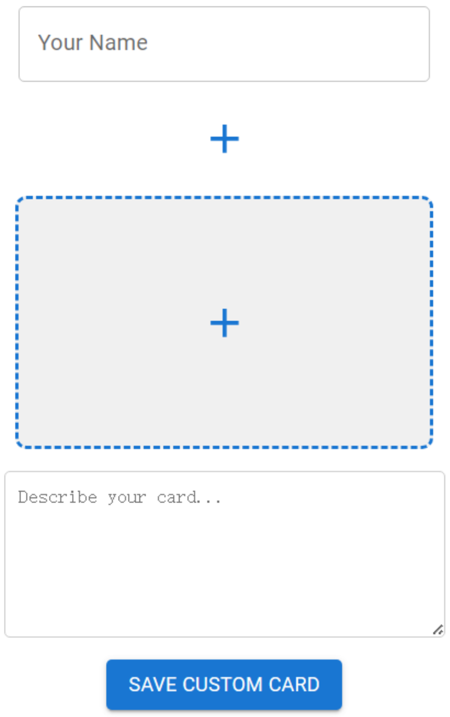


After logging in you will see the main page as follows:

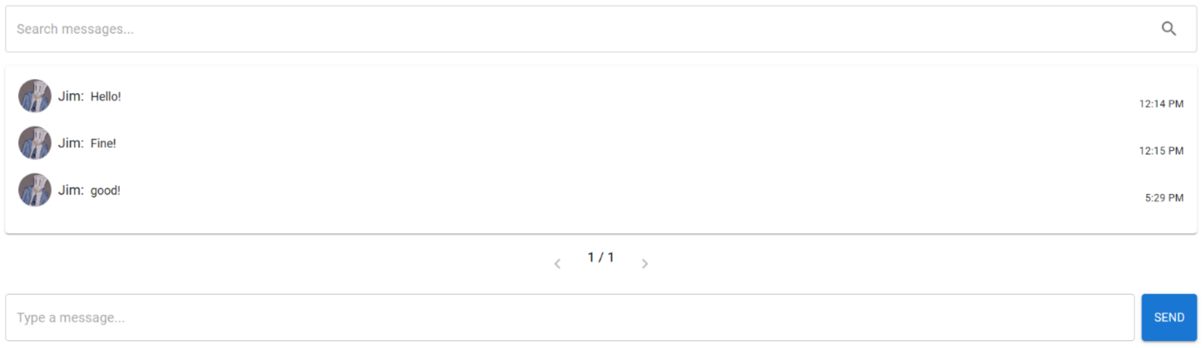


In the upper right corner of the main page is your avatar and username. By clicking on your avatar, you will be able to see your personal information in the sidebar on the right hand side of the page, including a larger image of your avatar, your username, a brief description and the date your account was created.

By clicking on the "Customize Card" button at the top left of the page, you can edit your personal information as well as your personal card. By default, the card displays your username and profile, and you can set additional descriptions and background images for the card. You can upload a new avatar by clicking on the "+" sign at the top of the card. After saving, the new avatar will be displayed in the navigation bar and in the sidebar where your profile is displayed. By clicking on the "+" sign at the bottom, you can set a background image for your card. You can also change your username and description, after saving they will be synchronized with your profile.



Each user card on the main page has a "like" and a "dislike" option, which you can select by swiping left (dislike), swiping right (like) or clicking on a button. If you like a user's card, you can enter a chat with that user. You can send a message to the user, but only if the user also likes your card before he/she sees the message and replies to you.



Jim: Hello! 12:14 PM
Jim: Fine! 12:15 PM
Jim: good! 5:29 PM

It should also be noted that, in order to facilitate the testing of the application, I directly write the user's authentication key (json-web-token) in the code (refer to the contents of server/config/passport.js). When actually installing and using the application, you may want to create an .env file and store your authentication key in it for better security.

4. Requirements and points

Requirements in the project instructions:

Feature	**Status (Done, partial-implemented, Not implemented)**	**Recommended points**
Basic features (as stated in the previous chapter) with well written documentation	Done	25
Utilization of a frontside framework, such as React, but you can also use Angular, Vue or some other	Done (React)	5
One can swipe to left or right to dislike or like the profile	Done	2
Use of a pager when there is more than 10 chats available	Done	2
Login with Facebook, Google, X or other accounts (use Passport.js)	Partial-implemented (I made the route and passport, but need an ClientID for OAuth 2.0)	1
Admin account with rights to edit all the users and comments and delete content/users (if a user is removed, all its chats should be removed too)	Not implemented	0
Test software for accessibility; can it be used only with keyboard / voice command? Can screen readers work with your application?	Not implemented	0
Provide a search that can filter out only those messages that have the searched keyword	Done	3
If match is being found the UI gives option to start chat immediately	Done (can send message immediately when like a card)	2
User profiles can have images which are shown on the main page and in the chat	Done (Need to refresh after sending messages)	3
User can click username and see user profile page where name, register date, (user picture) and user bio is listed	Done (Click the avatar instead)	2
Last edited timestamp is stored and shown within chat	Done	2
Translation of the whole UI in two or more languages	Not implemented	0

Create (unit) tests and automate some testing for example with https://www.cypress.io/ (at least 10 cases have to be implemented)	Done (../cypress/e2e)	5
Extra features		**Recommended points**
Users have customized personal cards. Users can view and like other users' cards.		3
Password strength verification is set, e.g. it should contain at least one uppercase letter, one lowercase letter, one number and one special symbol.		1
Users can exclude cards they don't want to see, and these cards will be removed from the user's page.		1
Users can log out and change to another account		1

Suggested points total: 48-50**.**

Github repository link for the codes: https://github.com/CyberBug2077-v/web-app.git