FindLove

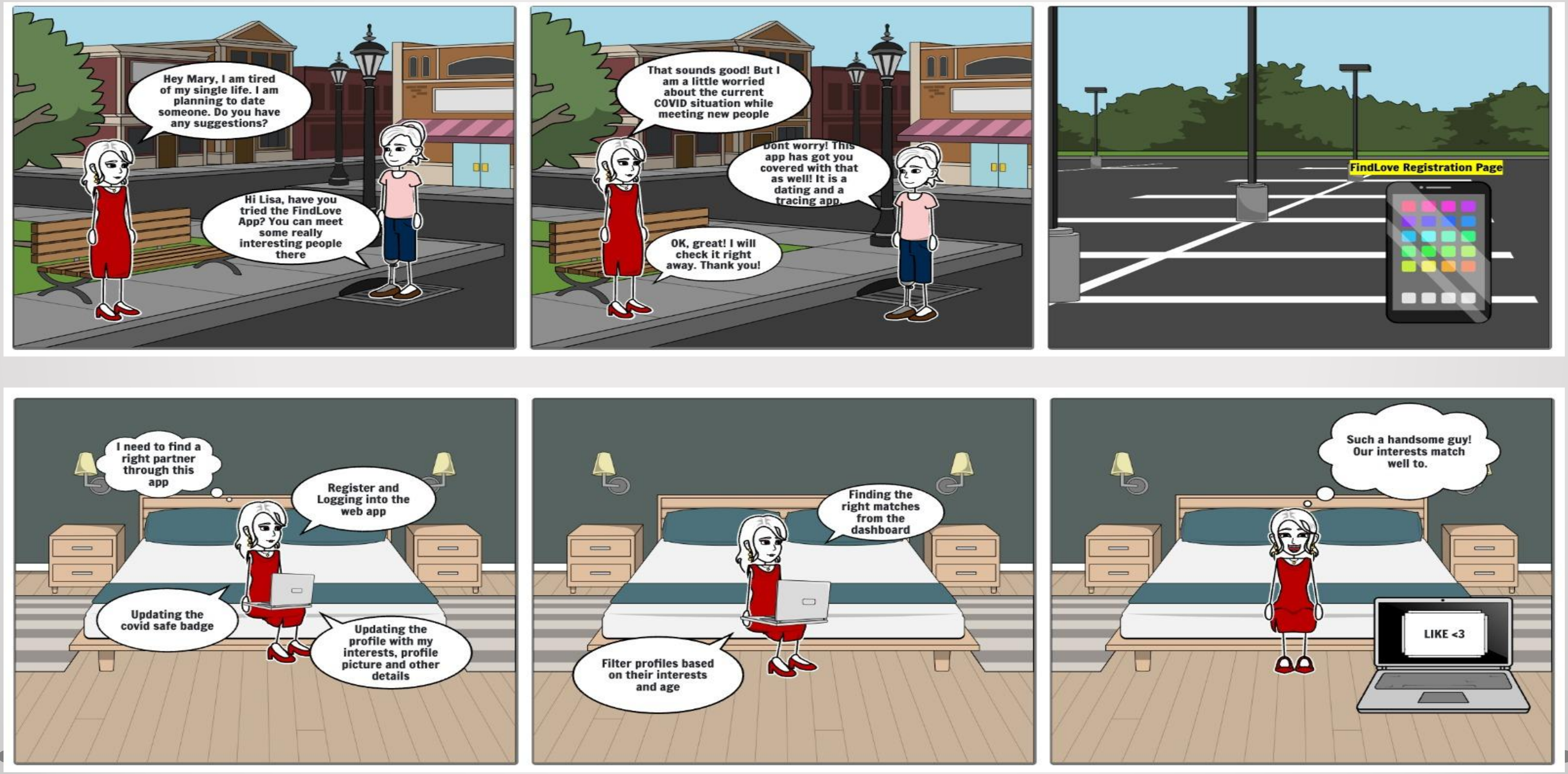
Application Overview

- Name of the application
  - “Find Love”

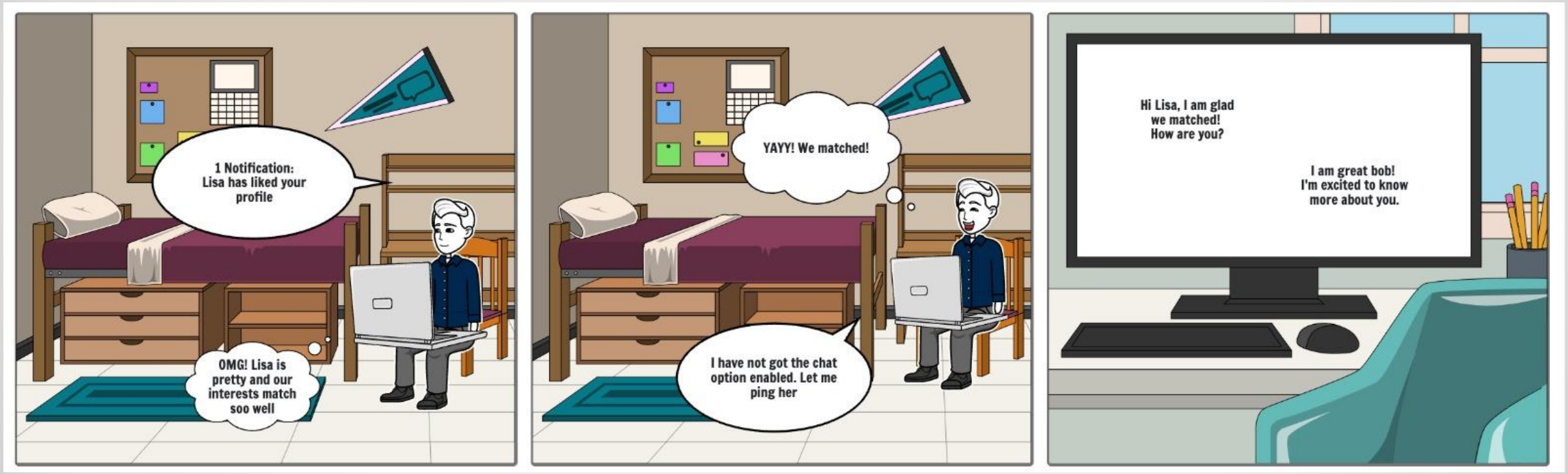
- Motivation of project
  - A privacy-by Design based dating application
  - Help individuals build friendship and locate compatible partners
  - Assess socially without the need to maintain social distance during COVID-19

- Goals of project
  - Finding a compatible match with people who share their interests
  - Self-assessment feature to evaluate whether they are at risk of contracting COVID-19
  - Security features

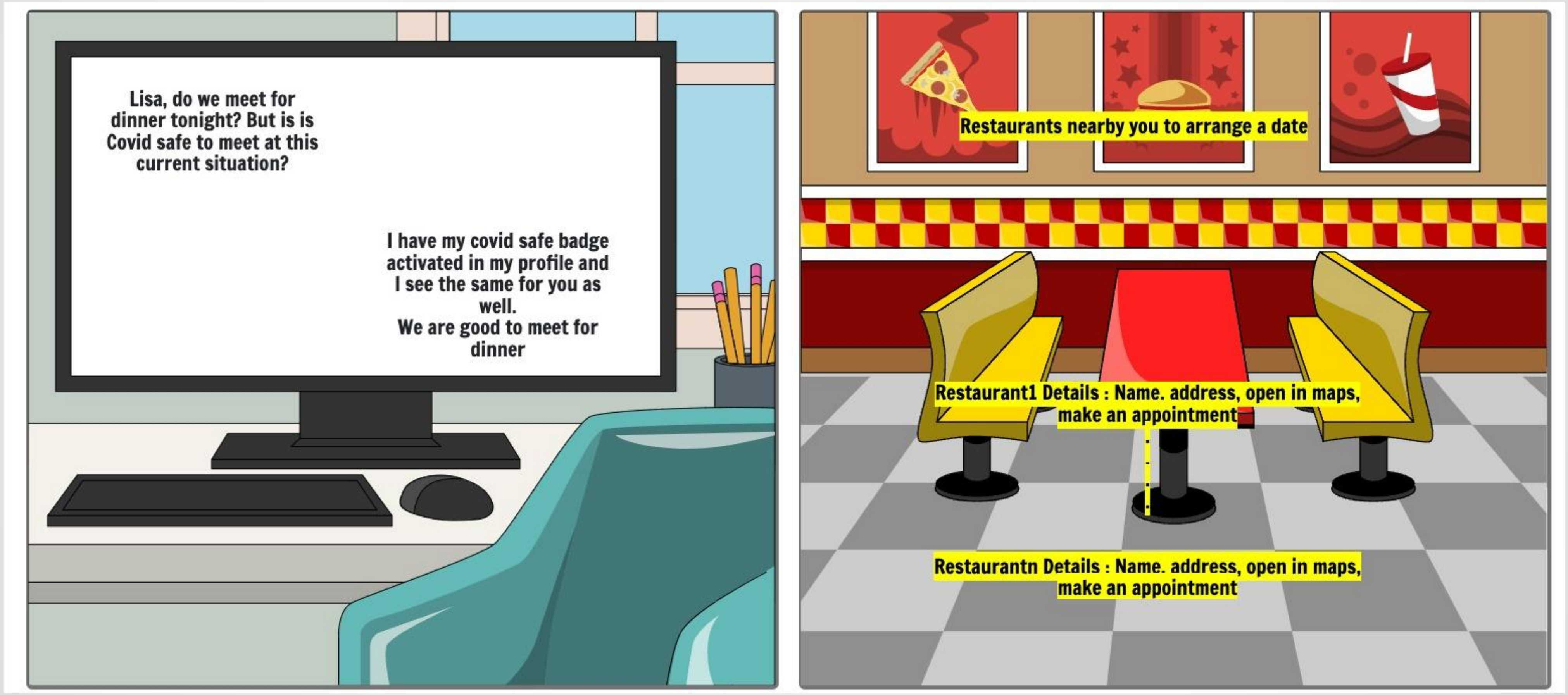
Application Functional Features


Hey Mary, I am tired of my single life. I am planning to date someone. Do you have any suggestions?
Hi Lisa, have you tried the FindLove App? You can meet some really interesting people there
That sounds good! But I am a little worried about the current COVID situation while meeting new people
Dont worry! This app has got you covered with that as well! It is a dating and a tracing app.
OK, great! I will check it right away. Thank you!
FindLove Registration Page
I need to find a right partner through this app
Register and Logging into the web app
Updating the covid safe badge
Updating the profile with my interests, profile picture and other details
Finding the right matches from the dashboard
Filter profiles based on their interests and age
Such a handsome guy! Our interests match well to.
LIKE <3

Application Functional Features


1 Notification:
Lisa has liked your profile
OMG! Lisa is pretty and our interests match soo well
YAYY! We matched!
I have not got the chat option enabled. Let me ping her
Hi Lisa, I am glad we matched! How are you?
I am great bob! I'm excited to know more about you.

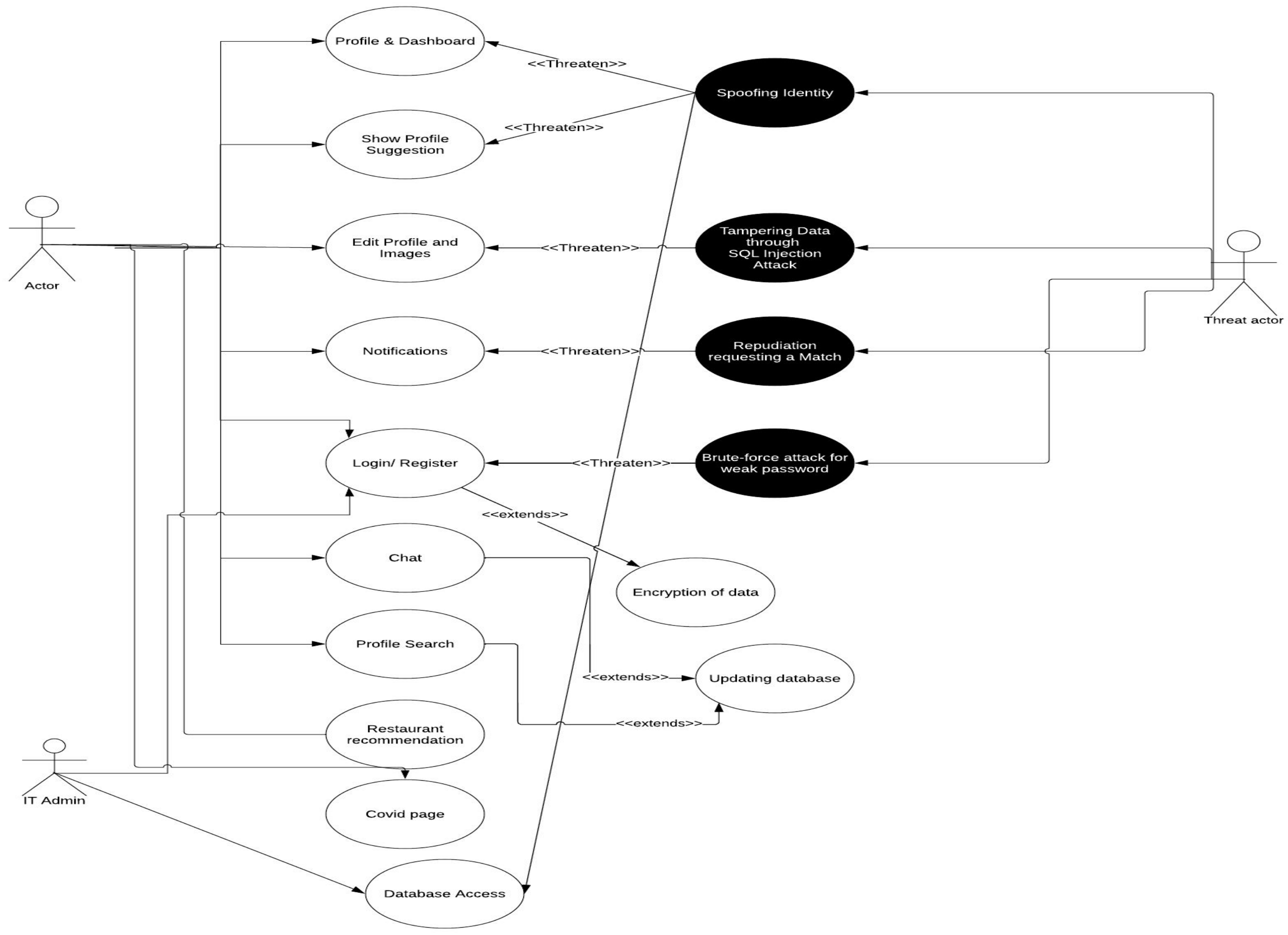
Application Functional Features


Lisa, do we meet for dinner tonight? But is is Covid safe to meet at this current situation?
I have my covid safe badge activated in my profile and I see the same for you as well. We are good to meet for dinner
Restaurants nearby you to arrange a date
Restaurant1 Details : Name. address, open in maps, make an appointment
Restaurantn Details : Name. address, open in maps, make an appointment

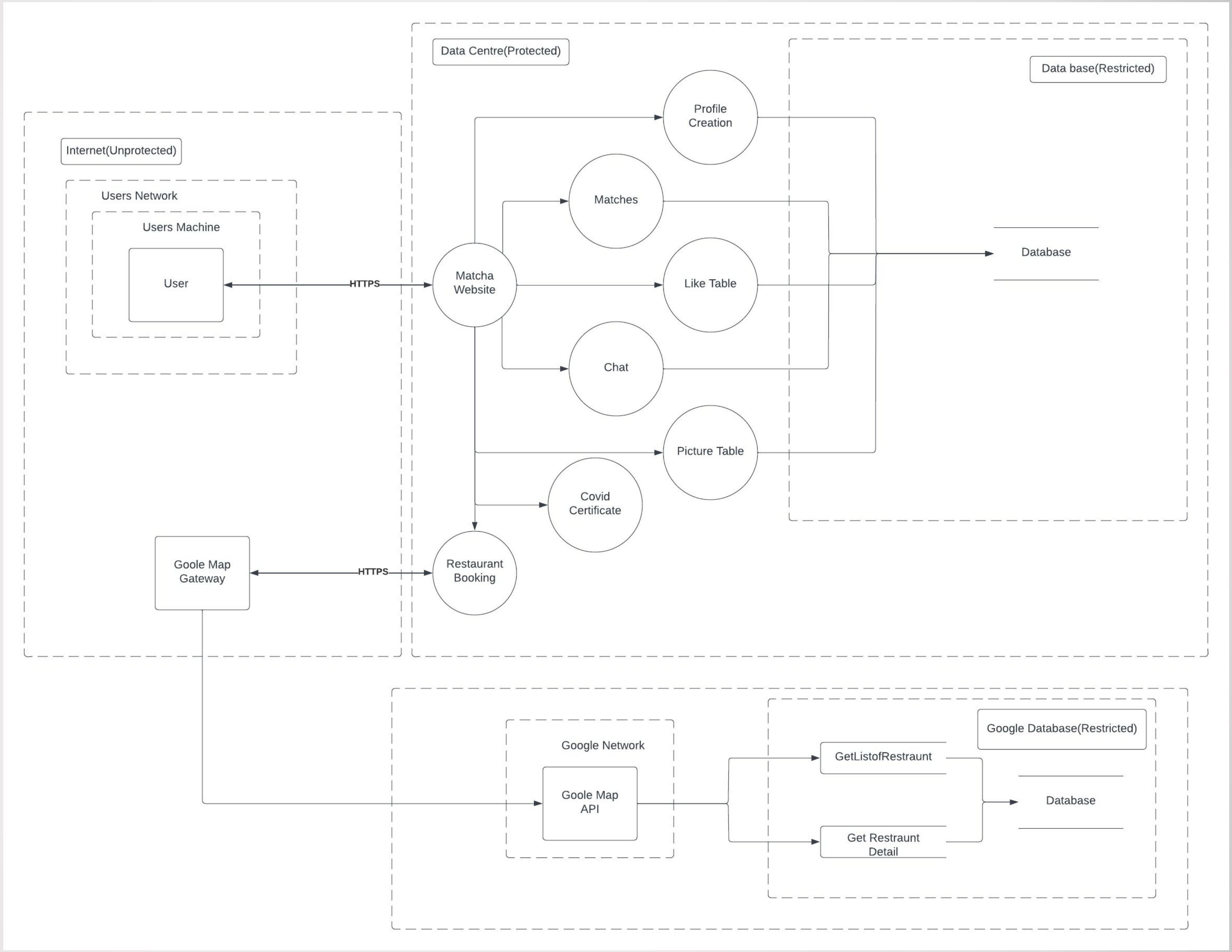
Security Requirements

- Cross-Site scripting (CWE - 79)
- Improper Authentication (CWE - 287)
- Clear text storage of Sensitive Information (CWE - 312)
- Rate Limiter (CWE - 770)
- SQL Injection (CWE - 89)
- Reliance on security through obscurity (CWE - 656)
- CORS (CWE - 346)

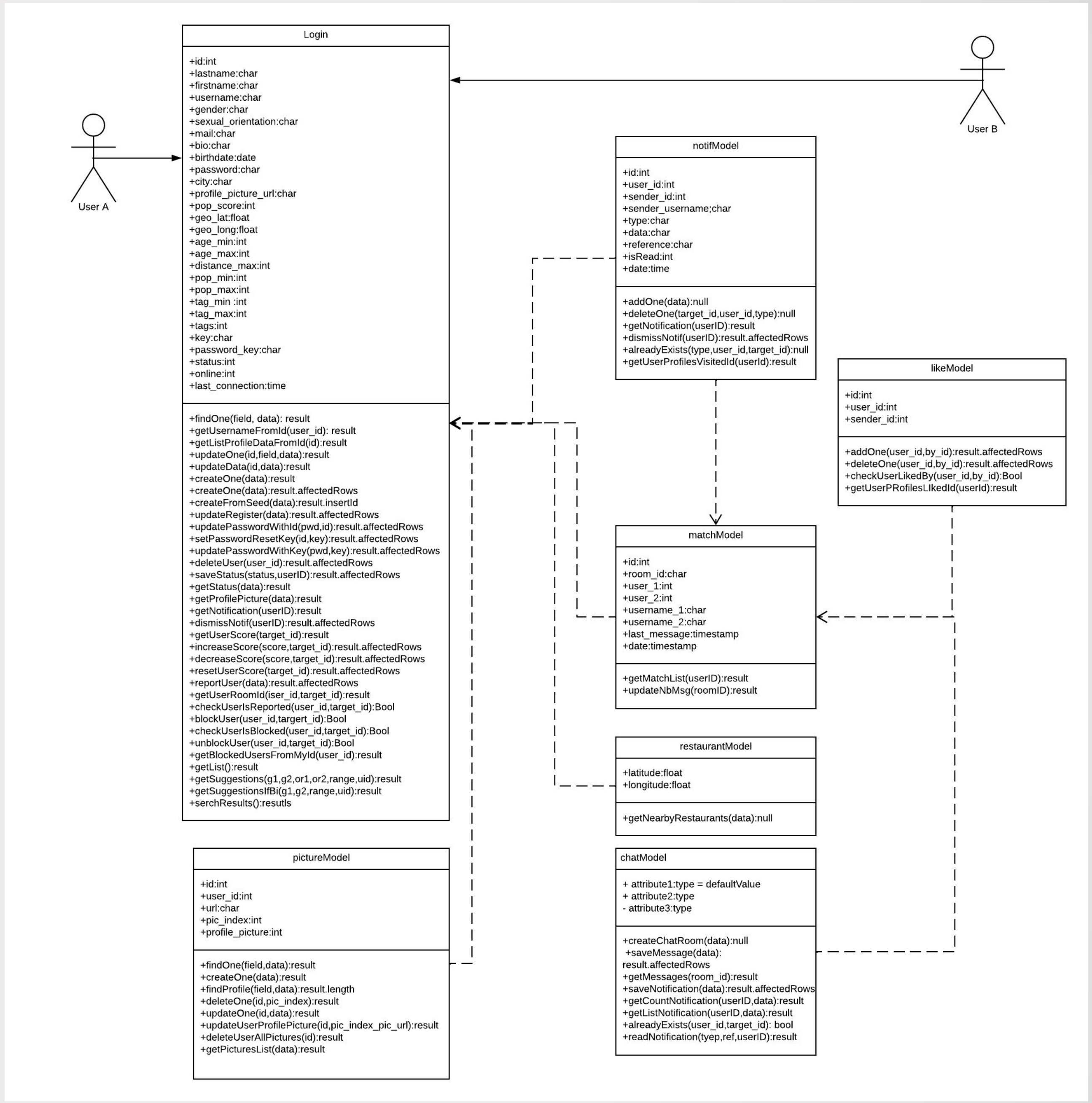
Use Case Diagram



Data Flow Diagram

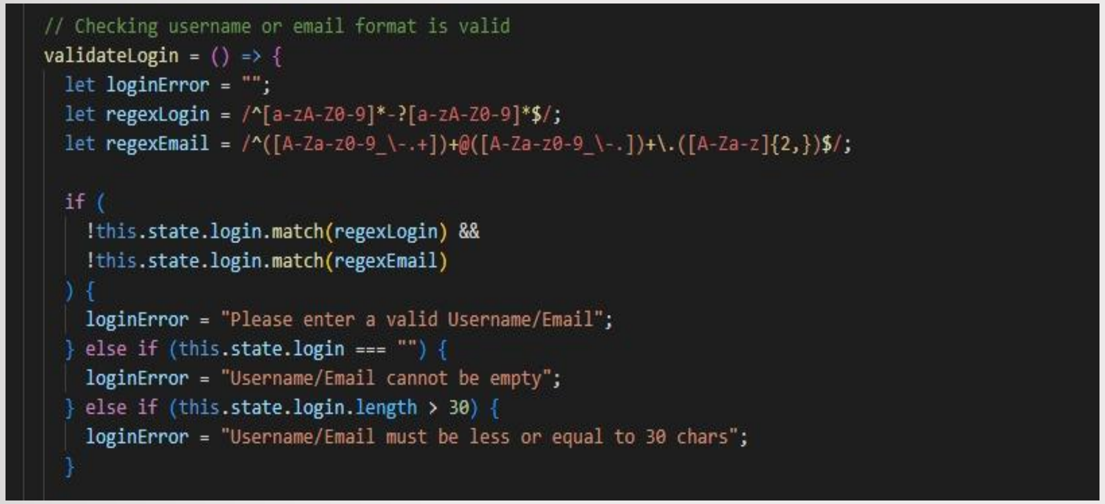


UML Diagram


User A
User B
Login
+id:int
+lastname:char
+firstname:char
+username:char
+gender:char
+sexual_orientation:char
+mail:char
+bio:char
+birthdate:date
+password:char
+city:char
+profile_picture_url:char
+pop_score:int
+geo_lat:float
+geo_long:float
+age_min:int
+age_max:int
+distance_max:int
+pop_min:int
+pop_max:int
+tag_min :int
+tag_max:int
+tags:int
+key:char
+password_key:char
+status:int
+online:int
+last_connection:time
+findOne(field, data): result
+getUsernameFromId(user_id): result
+getListProfileDataFromId(id):result
+updateOne(id,field,data):result
+updateData(id,data):result
+createOne(data):result
+createOne(data):result.affectedRows
+createFromSeed(data):result.insertId
+updateRegister(data):result.affectedRows
+updatePasswordWithId(pwd,id):result.affectedRows
+setPasswordResetKey(id,key):result.affectedRows
+updatePasswordWithKey(pwd,key):result.affectedRows
+deleteUser(user_id):result.affectedRows
+saveStatus(status,userID):result.affectedRows
+getStatus(data):result
+getProfilePicture(data):result
+getNotification(userID):result
+dismissNotif(userID):result.affectedRows
+getUserScore(target_id):result
+increaseScore(score,target_id):result.affectedRows
+decreaseScore(score,target_id):result.affectedRows
+resetUserScore(target_id):result.affectedRows
+reportUser(data):result.affectedRows
+getUserRoomId(iser_id,target_id):result
+checkUserIsReported(user_id,target_id):Bool
+blockUser(user_id,targert_id):Bool
+checkUserIsBlocked(user_id,target_id):Bool
+unblockUser(user_id,target_id):Bool
+getBlockedUsersFromMyId(user_id):result
+getList():result
+getSuggestions(g1,g2,or1,or2,range,uid):result
+getSuggestionsIfBi(g1,g2,range,uid):result
+serchResults():resutls
notifModel
+id:int
+user_id:int
+sender_id:int
+sender_username;char
+type:char
+data:char
+reference:char
+isRead:int
+date:time
+addOne(data):null
+deleteOne(target_id,user_id,type):null
+getNotification(userID):result
+dismissNotif(userID):result.affectedRows
+alreadyExists(type,user_id,target_id):null
+getUserProfilesVisitedId(userId):result
likeModel
+id:int
+user_id:int
+sender_id:int
+addOne(user_id,by_id):result.affectedRows
+deleteOne(user_id,by_id):result.affectedRows
+checkUserLikedBy(user_id,by_id):Bool
+getUserPRofilesLIkedId(userId):result
matchModel
+id:int
+room_id:char
+user_1:int
+user_2:int
+username_1:char
+username_2:char
+last_message:timestamp
+date:timestamp
+getMatchList(userID):result
+updateNbMsg(roomID):result
restaurantModel
+latitude:float
+longitude:float
+getNearbyRestaurants(data):null
pictureModel
+id:int
+user_id:int
+url:char
+pic_index:int
+profile_picture:int
+findOne(field,data):result
+createOne(data):result
+findProfile(field,data):result.length
+deleteOne(id,pic_index):result
+updateOne(id,data):result
+updateUserProfilePicture(id,pic_index_pic_url):result
+deleteUserAllPictures(id):result
+getPicturesList(data):result
chatModel
+ attribute1:type = defaultValue
+ attribute2:type
- attribute3:type
+createChatRoom(data):null
+saveMessage(data):
result.affectedRows
+getMessages(room_id):result
+saveNotification(data):result.affectedRows
+getCountNotification(userID,data):result
+getListNotification(userID,data):result
+alreadyExists(user_id,target_id): bool
+readNotification(tyep,ref,userID):result

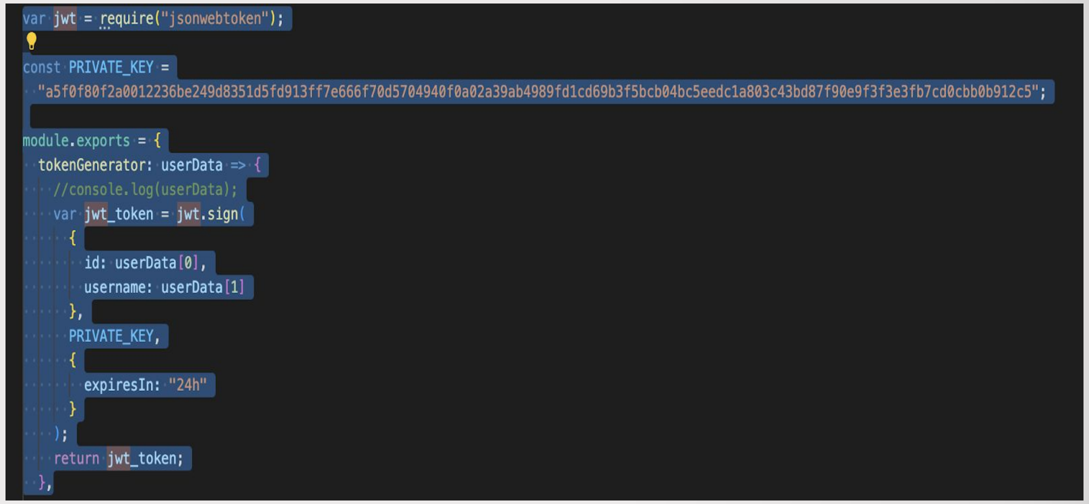
Security Implementation

- **XSS**



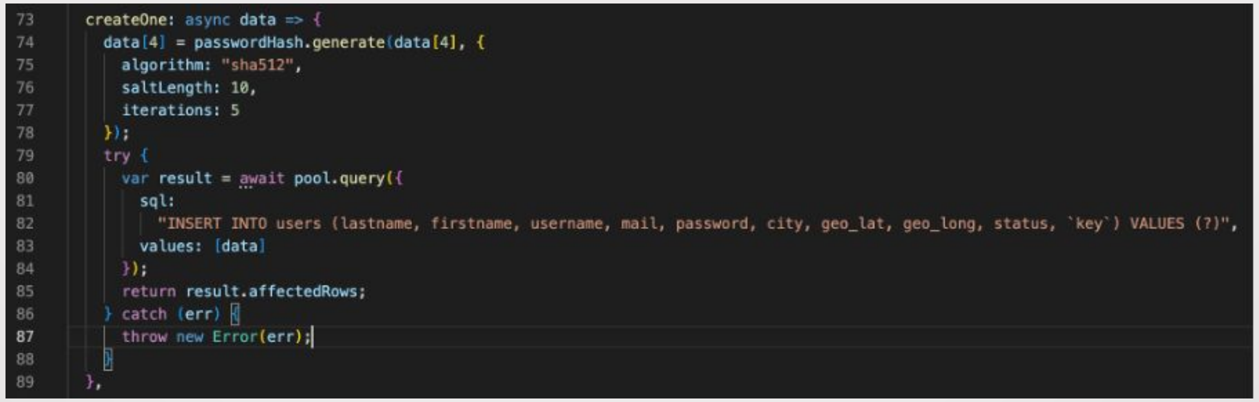
```
// Checking username or email format is valid
validateLogin = () => {
  let loginError = "";
  let regexLogin = /^[a-zA-Z0-9]*-?[a-zA-Z0-9]*$/;
  let regexEmail = /^([A-Za-z0-9_\-.+])+@([A-Za-z0-9_\-.])+\.([A-Za-z]{2,})$/;

  if (
    !this.state.login.match(regexLogin) &&
    !this.state.login.match(regexEmail)
  ) {
    loginError = "Please enter a valid Username/Email";
  } else if (this.state.login === "") {
    loginError = "Username/Email cannot be empty";
  } else if (this.state.login.length > 30) {
    loginError = "Username/Email must be less or equal to 30 chars";
  }
```

- **Improper Authentication**



```
var jwt = require("jsonwebtoken");

const PRIVATE_KEY =
  "a5f0f80f2a0012236be249d8351d5fd913ff7e666f70d5704940f0a02a39ab4989fd1cd69b3f5bcb04bc5eedc1a803c43bd87f90e9f3f3e3fb7cd0cbb0b912c5";

module.exports = {
  tokenGenerator: userData => {
    //console.log(userData);
    var jwt_token = jwt.sign(
      {
        id: userData[0],
        username: userData[1]
      },
      PRIVATE_KEY,
      {
        expiresIn: "24h"
      }
    );
    return jwt_token;
  },
```

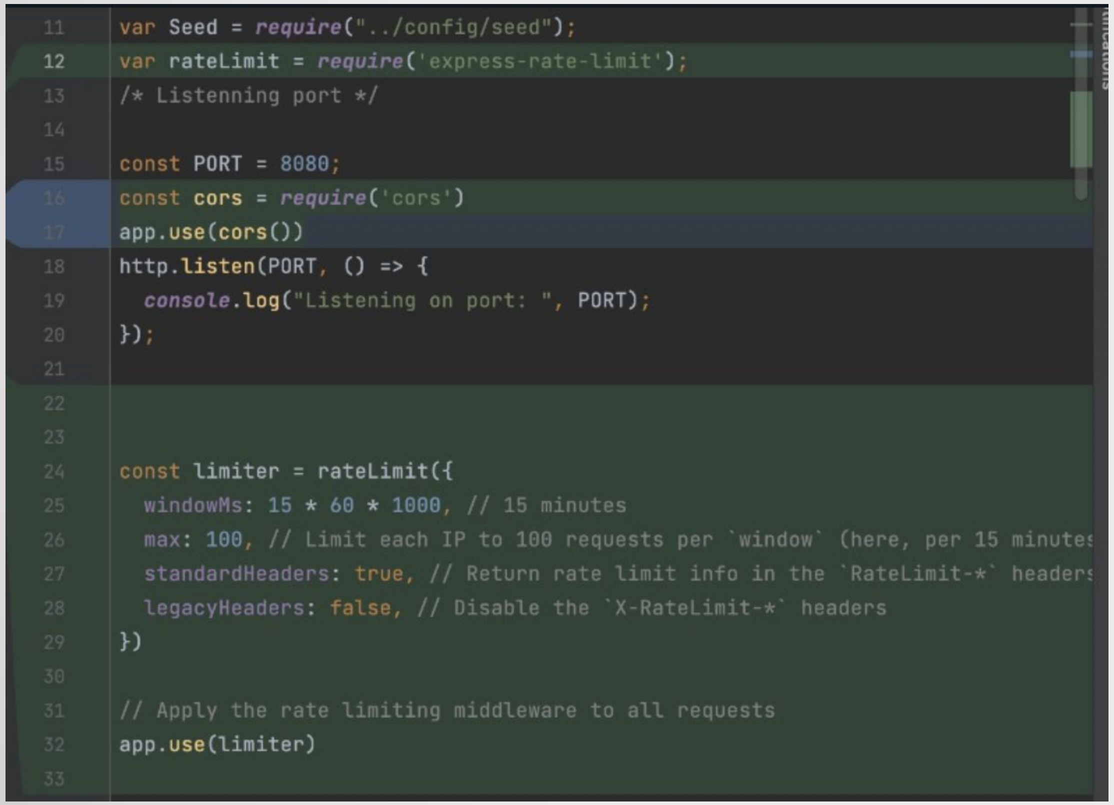
- **Clear text storage of Sensitive Information**



```
  createOne: async data => {
    data[4] = passwordHash.generate(data[4], {
      algorithm: "sha512",
      saltLength: 10,
      iterations: 5
    });
    try {
      var result = await pool.query({
        sql:
          "INSERT INTO users (lastname, firstname, username, mail, password, city, geo_lat, geo_long, status, `key`) VALUES (?)",
        values: [data]
      });
      return result.affectedRows;
    } catch (err) {
      throw new Error(err);
    }
  },
```

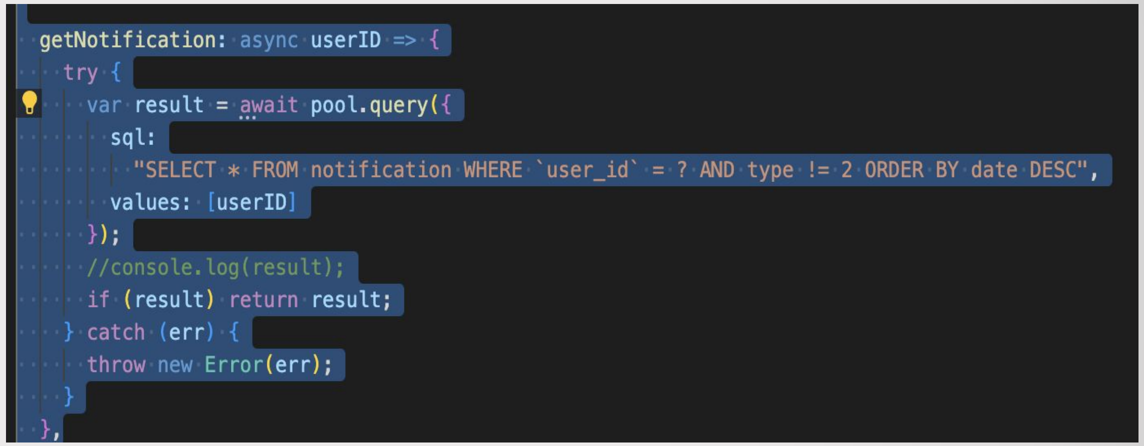
Security Implementation

- **Rate Limiter and CORS**



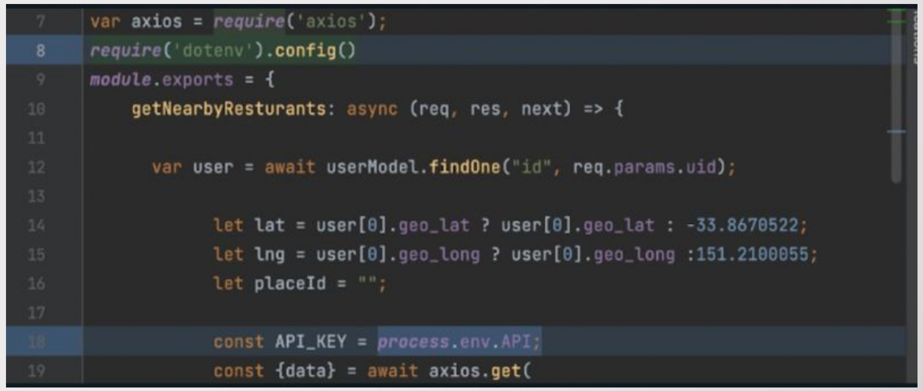
```
var Seed = require("../config/seed");
var rateLimit = require('express-rate-limit');
/* Listenning port */

const PORT = 8080;
const cors = require('cors')
app.use(cors())
http.listen(PORT, () => {
  console.log("Listening on port: ", PORT);
});



const limiter = rateLimit({
  windowMs: 15 * 60 * 1000, // 15 minutes
  max: 100, // Limit each IP to 100 requests per `window` (here, per 15 minutes
  standardHeaders: true, // Return rate limit info in the `RateLimit-*` headers
  legacyHeaders: false, // Disable the `X-RateLimit-*` headers
})

// Apply the rate limiting middleware to all requests
app.use(limiter)
```

- **SQL Injection**



```
getNotification: async userID => {
  try {
    var result = await pool.query({
      sql:
        "SELECT * FROM notification WHERE `user_id` = ? AND type != 2 ORDER BY date DESC",
      values: [userID]
    });
    //console.log(result);
    if (result) return result;
  } catch (err) {
    throw new Error(err);
  }
},
```

Removal of Hard-coded secrets •



```
var axios = require('axios');
require('dotenv').config()
module.exports = {
    getNearbyResturants: async (req, res, next) => {

      var user = await userModel.findOne("id", req.params.uid);

          let lat = user[0].geo_lat ? user[0].geo_lat : -33.8670522;
          let lng = user[0].geo_long ? user[0].geo_long :151.2100055;
          let placeId = "";

          const API_KEY = process.env.API;
          const {data} = await axios.get(
```

Future Work

- Mobile application (Google Play and Apple App Stores)
- Additional places besides restaurants such as clubs and galleries
- Machine learning solution (suggests user profiles on the dashboard based on past preferences)
- Notify users and include Covid hotspots when selecting a restaurant

Thank you.