Stamford International University

ITE-220

Web Development2

Instructor: Susanta Malakar

Htet Aung Wai : Student ID – 2402230001

Thanawat Khanijomdi: Student ID – 2306280005

Aanand Sirisukhaanand: Student ID – 2303170004

Executive Summary

MatchMaker is a responsive PHP web application that enables users to register with a profile photo, browse one profile at a time, like/dislike users, and form matches on mutual likes. The system implements secure session management and CSRF protection, persists data in MySQL (via PDO), saves user preferences using cookies, and includes a dynamic, AJAX-submitted contact form. The project demonstrates professional web practices across front end, back end, and database layers.

Table of Contents

Contents

1) Objectives

- Build a mobile first, responsive website with at least four pages.
- Persist in user, reaction, match, and message data in a relational database.
- Implement a dynamic form with AJAX submission and server-side persistence.
- Use sessions for authentication/state, and cookies for user preferences.
- Demonstrate secure coding (prepared statements, CSRF tokens, output escaping).
- Deliver maintainable, well-organized code and professional user experience.

---

2) Scope & Features

- **Registration with Photo Upload:** New users must provide name, age, bio, and a JPG/PNG profile photo to be visible to others.
- **Sequential Browsing (One at a Time):** Home shows exactly one "next unseen" complete profile—users advance with like/dislike.
- **Reactions & Matches:** Likes/dislikes are idempotent and stored. Mutual likes create a match.
- **Matches List:** A dedicated page list matches users with match time.
- **Preferences:** Users set a theme (light/dark) and max distance; stored as cookies and used to filter results.

- **Contact Form (AJAX):** Validates on client, posts asynchronously, saves to database, returns a success message without page reload.
- **Login/Logout:** Minimal login to create/use a session; logout clears the session cleanly.

---

3) Architecture & Technology Stack

- **Frontend:** HTML5, Bootstrap 5, custom CSS (Poppins), Vanilla JavaScript (Fetch API).
- **Backend:** PHP 8+ with PDO (MySQL).
- **State & Security:** PHP sessions (authentication + CSRF), cookies (preferences).
- **Database:** MySQL (via XAMPP); UTF-8 (utf8mb4).
- **Web Server:** Apache (XAMPP).

Design Approach:

- Separation of layout (header/footer) from page logic.
- Prepared statements for all DB writes; output escaping for all user content.
- Minimal, readable controllers (index/reaction/register/contact/etc.) with clear responsibilities.

---

4) Database Design

Entities:

- **users**(id, name, age, photo, distance_km, bio)
- **likes**(id, liker_id, liked_id, liked_at) — **UNIQUE(liker_id, liked_id)**
- **dislikes**(id, disliker_id, disliked_id, disliked_at) — **UNIQUE(disliker_id, disliked_id)**
- **matches**(id, user1_id, user2_id, matched_at) — **UNIQUE(user1_id, user2_id)** with (user1_id, user2_id) stored in ascending order
- **messages**(id, user_id, name, email, message, created_at)

Rationale:

- Separate **likes** and **dislikes** simplifies queries and ensures idempotency using unique pairs.
- **matches** is a denormalized convenience table for fast retrieval of mutual likes.
- **messages** captures contact form submissions; user_id is nullable for guests.
- Indexes on foreign keys improve performance on list and existence queries.

---

5) Application Pages & Flows

1. **Home – index.php:**
    - Displays one “next unseen” complete profile at a time (must have age, bio, photo).
    - Uses cookies for distance filtering; honors like/dislike history.
    - Injects a CSRF token into the page and posts it with Fetch to reaction.php.
2. **Register – register.php:**
    - Validates required fields (name, age 18–99, bio) and accepts JPG/PNG (≤3MB).
    - Stores uploaded photo under /uploads with a unique filename.
    - On success, sets $_SESSION['user_id'] and redirects to Home.
3. **Matches – matches.php:**
    - Lists mutual matches (joins against matches and users), newest first.
4. **Contact – contact.php:**
    - AJAX form (name, email, message).
    - Client-side validation (HTML5 + Bootstrap).
    - Server stores messages with timestamp; returns JSON for inline alerts.
5. **Preferences – preferences.php:**
    - Saves theme and max distance into cookies; Home uses them immediately.

6. **Login/Logout – login.php / logout.php:**
    - Minimal login sets $_SESSION['user_id'], regenerates session ID/CSRF, and resets the one-by-one feed pointer.
    - Logout clears and destroys the session.

---

6) Security & Data Protection

- **CSRF Protection:**

  CSRF token generated per session ($_SESSION['csrf']), embedded in the page, and verified via hash_equals in reaction.php.

- **SQL Injection Prevention:**

  All DB writes/reads use prepared statements with bound parameters.

- **XSS Mitigation:**

  All user-generated strings are echoed with htmlspecialchars.

- **Session Hardening:**

  session_regenerate_id(true) on login; no session data in URLs; logout destroys session and cookie.

- **Idempotent Writes:**

  INSERT IGNORE + unique constraints prevent duplicate likes/dislikes or matches.

---

7) Client-Side Validation & UX

- HTML5 required attributes and type validations.
- Bootstrap's validation states (was-validated, invalid-feedback).
- Clear call-to-action buttons, single-card layout to reduce cognitive load.
- Images sized and cropped with object-fit: cover for consistent presentation.
- Responsive navigation with a collapsing navbar on small screens.

---

8) Cookies & Sessions Usage

- **Sessions:**
  - $_SESSION['user_id']: identifies the logged-in user.
  - $_SESSION['csrf']: CSRF token for POST actions.
  - $_SESSION['last_seen_id']: tracks the last profile shown to enable one-by-one browsing.
- **Cookies:**
  - theme: "light" or "dark" — influences navbar/body style in header.php.
  - max_distance: numeric filter read on Home queries.

---

9) Implementation Details (Code Structure)

/matchmaker

```
├─ db.php              # PDO connection (UTF-8, exception mode)
├─ header.php          # Session start, head, navbar, styles/scripts
├─ footer.php          # Footer, Bootstrap bundle, main.js
├─ index.php           # Single-card feed; CSRF injection; like/dislike
├─ reaction.php        # Validates session + CSRF; inserts like/dislike; creates matches
├─ register.php        # Required fields; handles image upload to /uploads
├─ matches.php         # Lists mutual matches
├─ contact.php         # AJAX contact form to messages table
├─ preferences.php     # Saves theme + max distance cookies
├─ login.php           # Minimal login; resets feed pointer; regenerates CSRF
├─ logout.php          # Full session destroy
├─ main.js             # Fetch for contact + reactions, client helpers
├─ style.css           # Custom theme overrides
├─ schema.sql          # Database DDL + seed data
├─ /uploads            # User-uploaded profile images (runtime)
└─ /img                # Demo seed images (1.jpg … 8.jpg)
```

10) Setup & Execution (XAMPP)

1. Start **Apache** and **MySQL** in XAMPP.
2. Copy the project to C:\xampp\htdocs\matchmaker\.
3. In phpMyAdmin, import **schema.sql** into the database (e.g., mytestdb).
4. Update db.php with the correct DB name and port (default MySQL port is **3306**).
5. Access the site at: **http://localhost/matchmaker/index.php**.
6. Register a new account, then browse profiles one by one and test likes/dislikes.

---

11) Testing & Results

- **Registration:** Rejects missing fields, invalid age, wrong file type, or >3MB uploads.
- **Feed:** Displays one complete profile at a time; honors cookies and prior reactions.
- **Reactions:** CSRF verified; likes/dislikes saved; mutual likes produce a match; duplicate actions are ignored (idempotent).
- **Contact Form:** Submits via AJAX; server returns JSON; success and error messages rendered without reload.
- **Preferences:** Cookies persist across sessions; Home query respects max_distance.
- **Logout/Login:** Session resets as expected; new CSRF token issued on login.

12) Challenges & Resolutions

- **Session/CSRF failures on alternate ports (e.g., Live Server :3000):**

  Cause: cross-origin sessions not shared with PHP.

  Resolution: use Apache origin (http://localhost/...); inject and post CSRF from the same origin.

- **Schema mismatches/Foreign keys during early resets:**

  Cause: table structure evolved; inconsistent columns.

  Resolution: unified schema.sql, removed reset feature for production flow.

- **Incomplete profiles appearing in feed:**

  Resolution: query only profiles with non-empty photo/bio and non-null age.

---

13) Conclusion

MatchMaker fulfills the course requirements with a clean, secure, and responsive implementation. It showcases professional practices: normalized schema, secure stateful actions, AJAX interactions, and a focused user experience. The codebase is small, readable, and extensible for future enhancements like password-based auth, chat, or profile editing.

---

14) Appendix A: Schema (Overview)

users

id (PK), name, age, photo, distance_km, bio

likes

id (PK), liker_id (FK→users.id), liked_id (FK→users.id), liked_at, UNIQUE(liker_id, liked_id)

dislikes

id (PK), disliker_id (FK→users.id), disliked_id (FK→users.id), disliked_at,

UNIQUE(disliker_id, disliked_id)

matches

id (PK), user1_id (FK→users.id), user2_id (FK→users.id), matched_at, UNIQUE(user1_id,

user2_id)

messages

id (PK), user_id (FK→users.id, NULL OK), name, email, message, created_at

---

15) Appendix B: Screenshots

1. **Home – Single Profile View (Like/Dislike)**

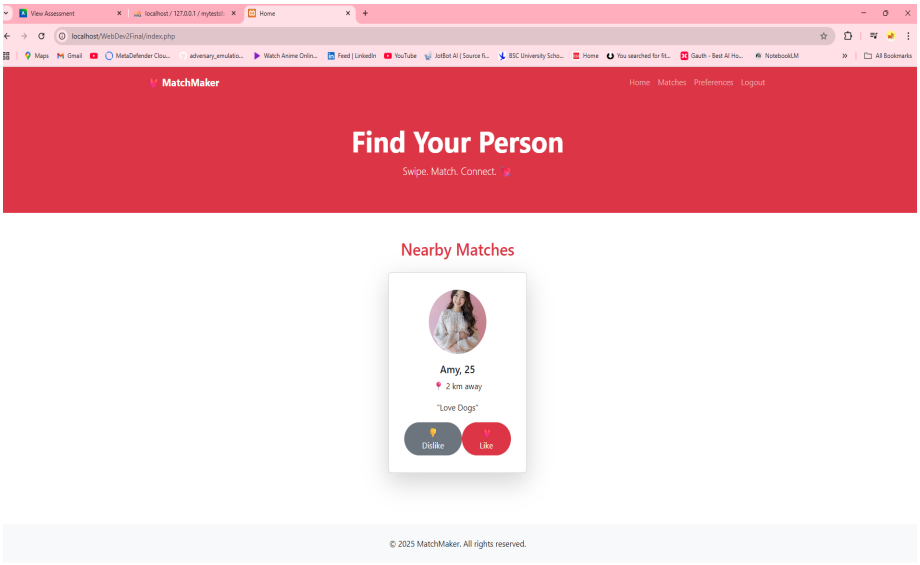


Figure 1 Single Profile View

2. **Register – Required Fields & Photo Upload**

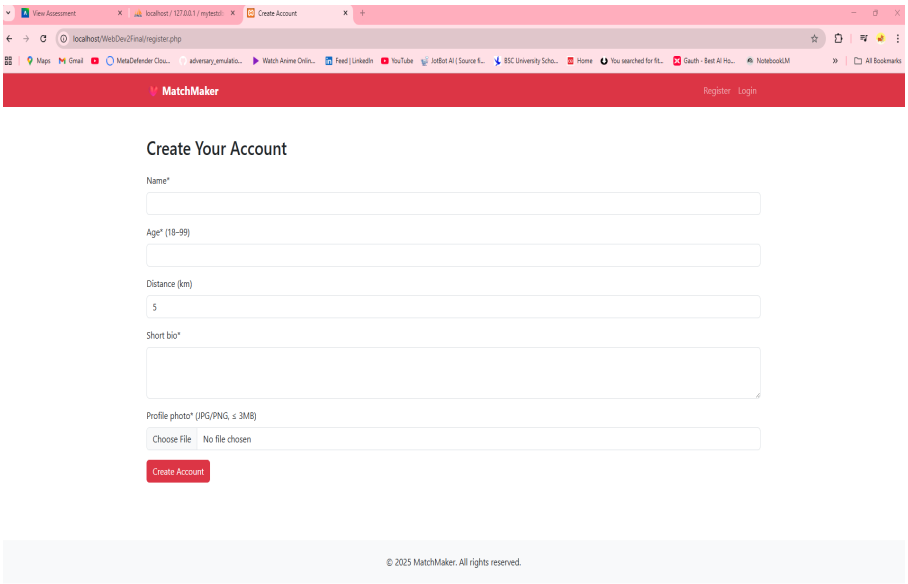


Figure 2 Register Account

3. Matches – List of Mutual Likes

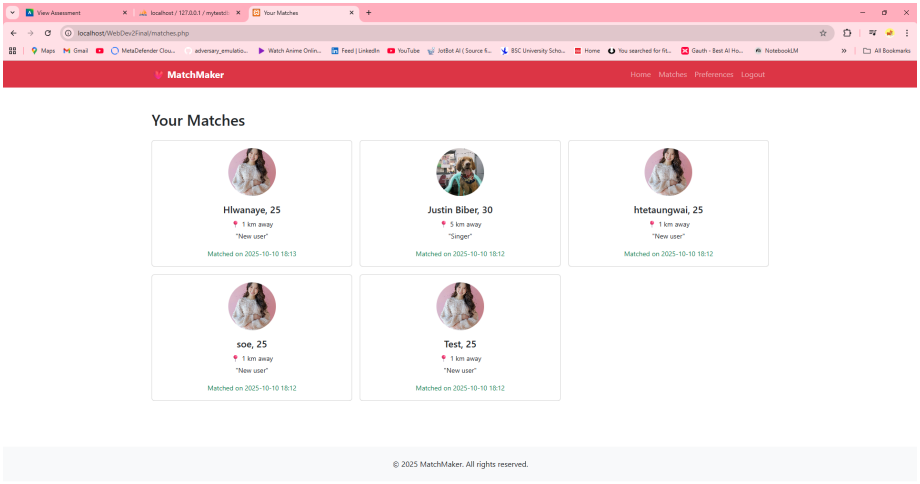


Figure 3 Mutual Likes

4. Contact – AJAX Success Message

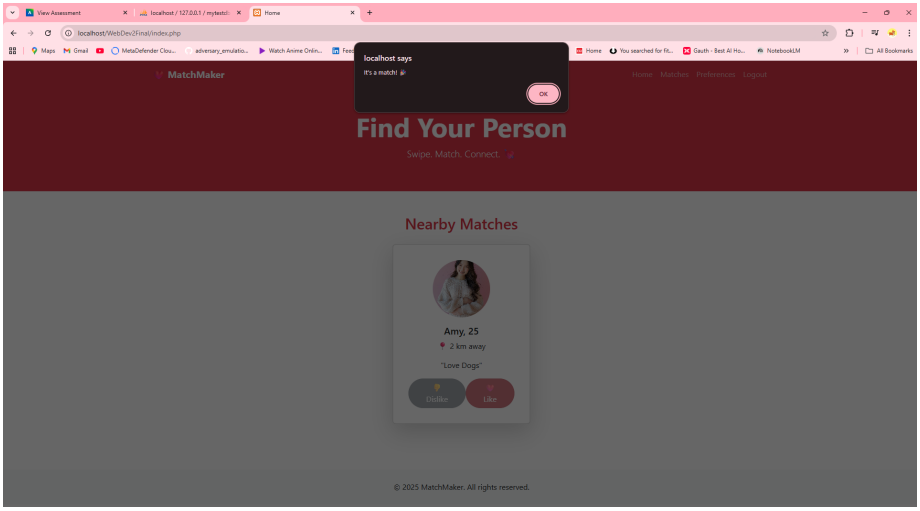


Figure 4 Ajax Success Message Match

5. **Preferences – Theme & Distance Cookies**

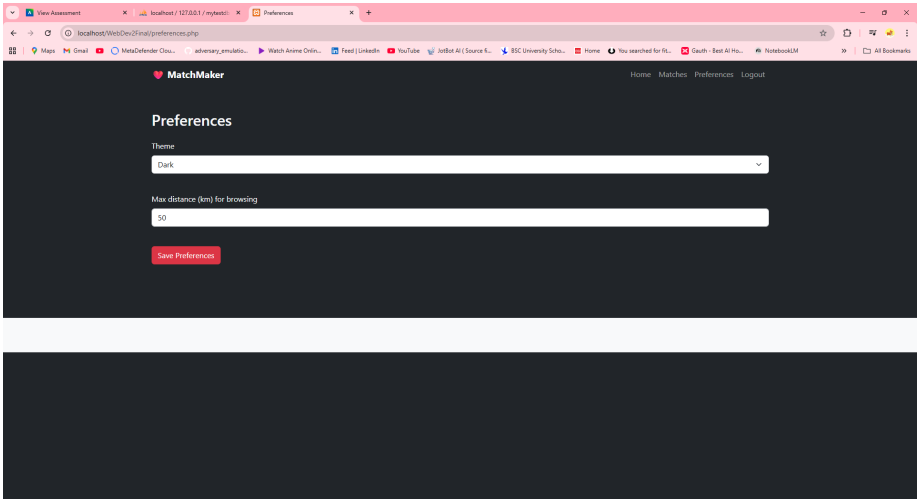
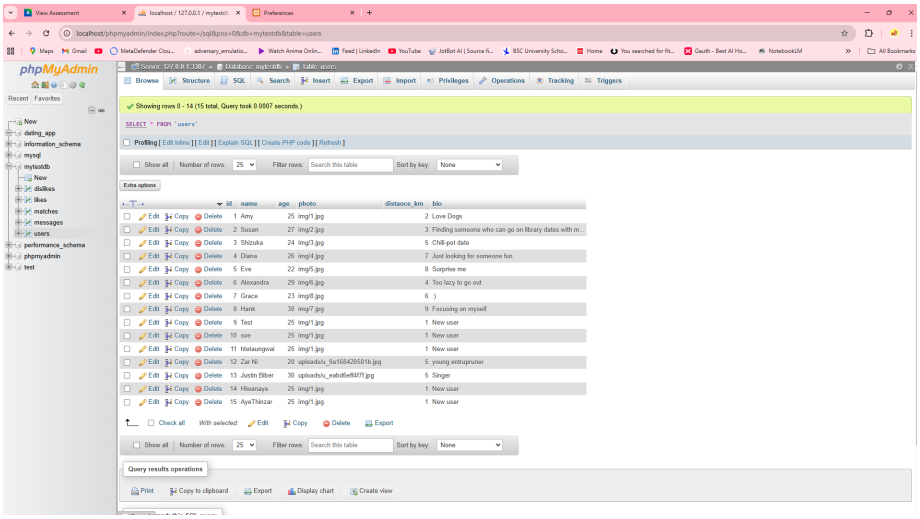


Figure 5 Changing Themes & Distance

6. **phpMyAdmin – Tables After Import**



Optional Future Work

- Password-based authentication with hashed passwords (e.g., password_hash).
- Profile edit page to update photo/bio post-registration.
- Real-time chat or notifications for matches.
- Image resizing/compression on upload to optimize storage and bandwidth.