Some documentation for Sprint 3

Sprint Planning Meeting (Meeting 1)

Date: 4 September 2025
Venue: Online

Agenda & Discussions:

- Discussed app features for Sprint 3 deliverables.
- Out of the four potential features, we agreed to present only *Improvements to Profile* and *Random Matchmaking*. We also agreed to present the *Chatting System* and *Cultural Explorer* page.
- **Chatting System Discussions:**
    - **Data Model & Firestore Collections:**
        - *profiles* collection for user info (username, avatar, country, languages, hobbies, etc.).
        - *matches* collection to represent pen pal relationships (two user IDs).
        - *chats* collection for conversations, with fields:
            - chatId (unique identifier from user IDs)
            - users (array of user IDs in the chat)
            - messages (array of message objects with sender, recipient, text, deliveryTime, etc.)
            - createdAt (timestamp).
    - **Matchmaking & Chat Creation:** Matches made via /api/matchmaking. When confirmed, a chat document is created and linked to both users.
    - **Messaging:** Messages sent via /api/chat/send, appended to chat's messages array.
    - **Retrieval:** Chats queried where user is a participant; messages displayed chronologically with pagination support.
    - **Moderation & Reporting:** Reports stored in a *reports* collection for admin review.

    - **Dashboard & Stats:** Total letters sent, active pen pals, countries connected, and recent activity.
    - **Real-time Updates:** Firestore listeners considered, though REST endpoints will be used primarily.
    - **Security & Privacy:** Only matched users can access chats; IDs used for access control.
    - **Extensibility:** Schema allows future features (favorites, reporting, attachments, emojis).
- **Cultural Explorer Discussions:**
    - Two sections: *country profiles* and *country facts*.
    - **Country Profiles:** Implemented via REST API (provides all needed information).
    - **Country Facts:** Team will collect predefined facts (food, tradition, etc.) manually.
        - Facts will be matched with countries by converting stored *timezones* in user profiles to countries, then querying the fact list.

Decisions Made:

- Focus on *Chatting System* and *Cultural Explorer* for Sprint 3 demo.
- Explorer page will have a section on country profiles (via REST API) and country facts (based on predefined dataset tied to user timezones).
- Chat system design finalized to rely on Firestore collections (*profiles, matches, chats, reports*).

Sprint (Meeting 2) – Designs for Implementation & Estimates

Date: 10 September 2025
Venue: In-person

Agenda & Discussions:

• Assigned **estimates** for user stories.
• Spoke about the idea behind new UI designs according to the stakeholder's request for better UI. Finalize and began implementing designs for the new pages:

- **Cultural Explorer page (new page)**
- **Profile (new improved design)**
- **Edit Profile (new and improved design)**
- **Chat system Page (new page)**
- **Dashboard (new and improved design)**

Decisions Made:

• Finalized designs and began implementation for profile-related pages.
• User story estimates completed.
• Updating database schema. According to what is presented in the database documentation.

Sprint (Meeting 3) – In-person Team Coding Session

Date: 19 September 2025
Venue: In-person

Agenda & Discussions:

- Checked progress on tasks across frontend and backend.
- Frontend and backend developers collaborated closely on implementations and UI.
- **Chatting System Implementation Progress:**
    - Setup of Firestore collections (*profiles, matches, chats, reports*) completed.
    - Implemented REST endpoints.

    - Frontend began integrating chat UI with backend endpoints, supporting message sending and retrieval.
    - Basic moderation flow (report creation) connected to *reports* collection.
- **Cultural Explorer Implementation Progress:**
    - Country profiles REST API successfully integrated.
    - Began creating a predefined dataset for country facts (food, traditions, etc.).
    - Implemented initial logic to map timezones from user profiles → countries → facts from dataset.

Decisions Made:

- Agreed on merging completed chat and explorer components with backend endpoints before the next sprint.
- Assigned debugging tasks for both chat messaging flows and explorer data retrieval.
- Set a deadline to complete unfinished chat and explorer features before the next meeting.

Sprint (Meeting 4) – Testing Strategy & User Feedback

Date: 25 September 2025
Venue: Online

Agenda & Discussions:

• Distributed **automation testing tasks** among team members.
• Discussed **user feedback testing methodology**:

- Continuation of our **formal feedback approach** using printed questionnaires.
- Users will download the app, use it, and answer structured questions.
- Users must **sign and date the forms** after completion.
  • Distributed printed forms to each member, who will collect responses individually.

Decisions Made:

• Automation testing responsibilities assigned.
• Continued with our formalized questionnaire-based feedback system.
• Distributed feedback forms and confirmed collection process among team members.

Stakeholder Interaction Meeting
Date: 23 September 2025
Venue: Online

Agenda & Discussions:
• Presented the client with the new and improved design.
• Provided an overview of deliverables completed so far.
• Asked follow-up questions related to the project rubric:
– Documentation section: Clarified expectations for API documentation format.
– Database: Confirmed that additions should be made directly to the schema.
– Testing: Asked if all tests (Sprint 3 tests, previous tests, and planned tests) should be documented.

Decisions Made:
• Client confirmed that API documentation should follow a clear and structured format.
• Client agreed that new changes should be added to the existing database schema.
• Client instructed the team to document all tests, including Sprint 3, past, and future tests.

plan to run for sprint 3, and test we plan on running in the future.) And under API implenetation, we asked if we should prove that our api is publically available for external usage, he said prove that for at least 1 endpoint, He suggested swagger, but said his not forcing us to use it.