# Board market

A place where people can upload their boards (Surfboards / Skateboards / Snowboards etc.) to a database. Specific details (brand, model, litres, length, condition etc) about the board will be entered when uploading using the form and then people will be able to search for boards in their area using the google maps API to filter based on proximity.

API - Google maps/ Geocoding

## 1. What goal will your website be designed to achieve?

The website will help connect surfers to find the perfect board for them. Currently most boards are sold on Facebook marketplace. This process involves adding the picture and a description however most of the time it is very hard to search through.

On Facebook marketplace you have to scroll through hundreds of boards to find one which is right for you which is time consuming and inefficient. Often there isn't much information on the boards listed on there and many boards can sit on there for months without finding a buyer.

The Board Market app will aim to connect buyers, sellers, lessors and lessees by acting as a platform where sellers can add their boards to our database and have them listed with ease. They will add the boards to the site by filling out a form.

Buyers will then be able to browse boards located close to where they live and also specify other properties to filter their search and find boards that would suit them. Once a buyer finds a board they like, they will then be able to send a message to the seller, which will appear in their inbox on the site as well as the possibility of a notification being sent to their email address.

The app would be able to notify buyers if a board is uploaded to the site that matches their saved search criteria. They can just leave the search online and not have to keep checking the website every day.

## 2. What kind of users will visit your site? In other words, what is the demographic of your users?

The platform is initially geared towards surfers, particularly focusing on young, health-conscious males. However, it should be an inclusive space, welcoming users of all ages and backgrounds. The goal is to build a diverse community centred around a shared passion for boardsports, ensuring accessibility and a sense of belonging for everyone involved.

# 3. What data do you plan on using? You may have not picked your actual API yet, which is fine, just outline what kind of data you would like it to contain.

## Mandatory form inputs:

- Asking price
- Board manufacturer
- Board length
- Condition (New, Like-New, Great, Good, Fair, Bad)
- Sell/Rent
- Location
- Delivery (No-Pickup only, Yes-Local, Yes-National)

## Recommended form inputs:

- Board model
- Board width
- Board depth
- Volume in litres
- Surfer level
- Extras included
- Extra details

## Database setup:

### Users:

- user_id (Primary Key)
- username
- email
- user_password
- created_at
- updated_at

## Boards:

- board_id (Primary Key)
- user_id (Foreign Key referencing Users)
- asking_price
- board_manufacturer
- board_length (Total length in inches)
- condition
- sell_or_rent
- board_location
- delivery_options
- brand
- model
- width
- depth
- volume_litres
- surfer_level
- extras
- extra_details
- created_at
- updated_at

## Messages:

- message_id (Primary Key)
- sender_id (Foreign Key referencing Users)
- receiver_id (Foreign Key referencing Users)
- board_id (Foreign Key referencing Boards)
- message_content
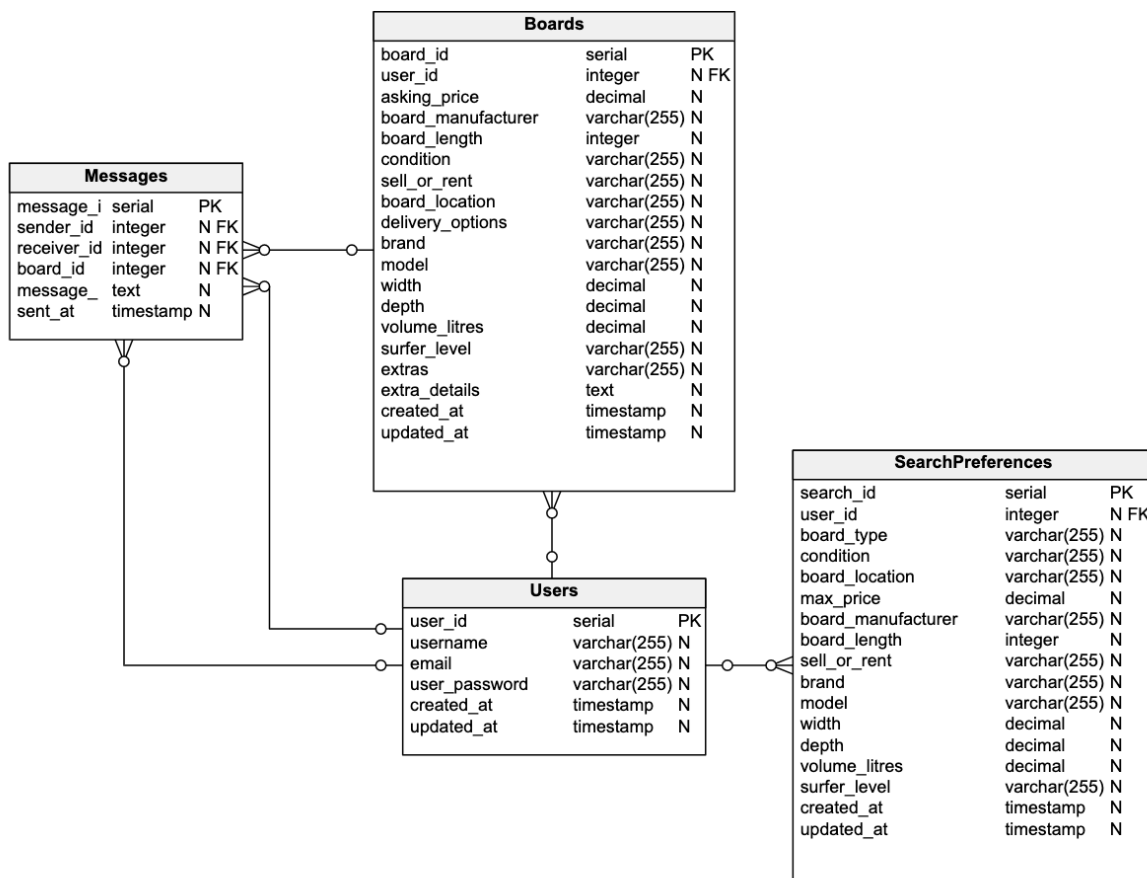- sent_at

## SearchPreferences:

- search_id (Primary Key)
- user_id (Foreign Key referencing Users)
- board_type
- condition
- board_location
- max_price
- board_manufacturer
- board_length (Total length in inches)
- sell_or_rent
- brand
- model

- width
- depth
- volume_litres
- surfer_level
- created_at
- updated_at

# 4. In brief, outline your approach to creating your project (knowing that you may not know everything in advance and that these details might change later).

- Make a schema showing the relationships between the table
- Create database on SQL and create tables for the data to be stored in
- Create the templates and routes to navigate to the pages
  - Create landing page which finds out if the user wants to:
    - sell your board
    - buy a board
    - rent your board
    - rent a board
  - Display the available boards on a board search page with a navbar to apply the filters
  - Board display page for when a board is selected
  - User profile page to see their boards displayed
  - Inbox page to view your messages with other users
  - Help page
  - Terms of service and privacy policy
- Create forms - WTForms
  - Signup form
  - Login form
  - Add board form
  - Search form
- Find out how to link google maps API to the location of users and be able to filter results based on proximity using spatial indexing and geolocations
- User authentication to protect accounts and data
- Create tests

## a. What does your database schema look like?

**Boards**

| | | |
|---|---|---|
| board_id | serial | PK |
| user_id | integer | N FK |
| asking_price | decimal | N |
| board_manufacturer | varchar(255) | N |
| board_length | integer | N |
| condition | varchar(255) | N |
| sell_or_rent | varchar(255) | N |
| board_location | varchar(255) | N |
| delivery_options | varchar(255) | N |
| brand | varchar(255) | N |
| model | varchar(255) | N |
| width | decimal | N |
| depth | decimal | N |
| volume_litres | decimal | N |
| surfer_level | varchar(255) | N |
| extras | varchar(255) | N |
| extra_details | text | N |
| created_at | timestamp | N |
| updated_at | timestamp | N |

**Messages**

| | | |
|---|---|---|
| message_i | serial | PK |
| sender_id | integer | N FK |
| receiver_id | integer | N FK |
| board_id | integer | N FK |
| message_ | text | N |
| sent_at | timestamp | N |

**Users**

| | | |
|---|---|---|
| user_id | serial | PK |
| username | varchar(255) | N |
| email | varchar(255) | N |
| user_password | varchar(255) | N |
| created_at | timestamp | N |
| updated_at | timestamp | N |

**SearchPreferences**

| | | |
|---|---|---|
| search_id | serial | PK |
| user_id | integer | N FK |
| board_type | varchar(255) | N |
| condition | varchar(255) | N |
| board_location | varchar(255) | N |
| max_price | decimal | N |
| board_manufacturer | varchar(255) | N |
| board_length | integer | N |
| sell_or_rent | varchar(255) | N |
| brand | varchar(255) | N |
| model | varchar(255) | N |
| width | decimal | N |
| depth | decimal | N |
| volume_litres | decimal | N |
| surfer_level | varchar(255) | N |
| created_at | timestamp | N |
| updated_at | timestamp | N |

## b. What kinds of issues might you run into with your API?

Geocoordinates Accuracy:
Be mindful of variations in geocoordinate accuracy from the Google Maps API. Users might have imprecise location data, leading to occasional inaccuracies. Consider implementing additional validation checks for data reliability.

Handling Large Datasets:
Efficiently manage large datasets for proximity-based searches. Optimise queries and responses to maintain reasonable server load and prevent slow response times.

Spatial Indexing in PostgreSQL:
Research spatial indexing in PostgreSQL to optimise location-based searches. Apply this knowledge to create effective spatial indexes for geo coordinates in our database.

Storage and Retrieval Efficiency:
Optimise storage and retrieval of geo coordinates in the database. Pay attention to data types, indexing strategies, and overall database schema for performance improvements.

Haversine Formula Considerations:
Implement the Haversine formula for distance calculations. Keep in mind its assumptions, such as a spherical Earth, and be aware of potential inaccuracies for long distances or locations near the poles.

API Usage Limits and Costs:
Understand Google Maps API usage limits and pricing policies. Regularly monitor usage to avoid exceeding free tiers and incurring unexpected costs. Adhere to usage policies to prevent service disruptions.

User Privacy Measures:
Inform users about the collection and use of their location data. Ensure their consent and implement privacy measures, such as using approximate addresses, to address privacy concerns.

API Key Security:
Safeguard the Google Maps API key. Implement secure storage practices and usage to prevent unauthorised access. Regularly review security measures to maintain the integrity of the key.

## c. Is there any sensitive information you need to secure?

### User Credentials:

Ensure the secure handling of sensitive information, including user emails and passwords. Implement robust encryption and storage practices to safeguard user login credentials.

### Data Protection Agreement:

Design a user-friendly data protection agreement form. This form should clearly outline the purposes for which user information will be used, ensuring compliance with data protection regulations. Users should explicitly agree to sharing their information and being contacted.

### Email Privacy:

Prioritise user email privacy by hiding email addresses within the platform. Establish an intermediary email system for user communication. This not only shields user emails from direct exposure but also adds an extra layer of security.

### Location Privacy:

Consider implementing location privacy measures. Instead of precise addresses, explore the option of using approximate locations to protect user privacy. This approach maintains the usability of location-based features while mitigating potential privacy concerns.

## d. What functionality will your app include?

### User Registration and Authentication:

- Users can create accounts and log in.
- Passwords and emails should be securely stored and encrypted.

### Board Listing:

- Sellers can add their boards to the database by filling out a form.
- Mandatory details like asking price, board manufacturer, length, condition, sell/rent, location, and delivery options will be included.
- Recommended details like board model, width, depth, volume in litres, surfer level, extras, and additional details can also be provided.
- Each board listing will have a unique board_id.

### Board Search:

- Buyers can browse boards located close to their area.
- Filtering options based on various parameters, including proximity using the Google Maps API.
- The app can notify buyers if a new board matching their saved criteria is listed.

### Messaging System:

- Users can send messages to sellers regarding specific boards.
- Messages will be stored, and notifications can be sent to users when they receive a new message.

### User Inbox:

- Users can view and manage their messages in an inbox on the site.
- Notifications can also be sent via email.

### User Preferences and Alerts:

- Users can save search criteria, and the app will notify them if a board matching their preferences is listed.
- Preferences might include if they want to be contacted by email or not and also what they want to be notified about; new messages in the inbox or new boards appearing in their search.

### Security and Privacy Measures:

- Secure handling of sensitive information such as user email and password.
- Data protection agreement to allow sharing and contact.
- An intermediary email system for user contact.
- Consider using approximate addresses rather than precise ones for privacy.

### Spatial Indexing:

- Implement spatial indexing in the database for efficient location-based searches.

### Google Maps Integration:

- Link Google Maps API to user locations.
- Enable filtering based on proximity using spatial indexing and geolocations.

### Forms and Validation:

- Use WTForms for creating and validating forms like sign-up, login, add board, and search.

### Testing:

- Implement test cases to ensure the functionality and security of the application.

## e. What will the user flow look like?

### Landing Page:

Users can choose to either sell, buy, or rent a board.

### User Registration/Login:

Users need to create an account or log in to access the features.

### Board Listing:

Sellers fill out a form to add a board for sale or rent.

### Board Search:

Buyers can search for boards using various filters, including location.

### Messaging:

Users can send messages to sellers regarding specific boards.

### Inbox:

Users can view and manage their messages.

### Preferences and Alerts:

Users can save search criteria and receive notifications for matching boards.

### Help Page:

A page providing information and assistance.

## f. What features make your site more than CRUD? Do you have any stretch goals?

### Messaging System:

The inclusion of a messaging system makes the site more interactive, facilitating communication between buyers and sellers.

### Notification System & User Preferences:

Enable users to receive personalised alerts for new board listings and messages. Users can save specific search criteria, including location and board preferences, and receive timely notifications within the app or via email. This comprehensive notification system enhances user engagement by providing tailored updates and streamlining the board discovery process.

### Privacy Measures:

Implementing measures to protect user information and ensuring secure contact methods enhances the overall security and privacy of the platform.

## Stretch Goals:

### User Reviews/Ratings:

Implement a system where users can leave reviews or ratings for boards and sellers, providing valuable feedback to the community.

### Social Media Integration:

Allow users to share their listings or find boards through social media platforms, expanding the reach of the app.

### Advanced Search Filters:

Provide more advanced search filters, such as specific board features, brand preferences, or board history.

### Mobile App:

Develop a mobile application for a more convenient and on-the-go experience for users.

## Currency Conversion:

Integrate a currency conversion feature to dynamically convert and display asking prices in the user's preferred currency, enhancing the global usability of the platform.

## Advertising Space:

Explore the incorporation of ad spaces on the site to generate additional revenue. Allow businesses or relevant partners to place advertisements, providing a potential income stream for the platform.