

# PinPoint

CSC1118 - Group Interview



# Demo Overview

- What is PinPoint?
- Business Case
- Technical Delivery
- Future Planning
- Contributions
- Prototype Showcase



# Team Overview

- Collaborative two-person development team;
  - Robert Maloney
  - Michael Beirne-Ponomarev
- Shared responsibility across design, development, and business strategy.
- Incremental workflow with regular check-ins with supervisor.
- Balanced skill sets: frontend, backend, UX, and commercial focus.



# What is PinPoint?

- Built around a live interactive map.
- Social and business event-planning app.
- Combines event creation, RSVP, chat, and discovery.



# Problem Statement

- Event planning is fragmented across multiple platforms.
- Group coordination is often confusing and inefficient.
- Local events are hard to discover without targeted search.



# Solution

- Centralised platform for event creation, invites, and chat.
- Interactive map to discover and join events.
- Custom visibility: public, private, or friends-only.
- Real-time group chats linked to each event.



# Key Features

- Event pin creation & RSVP.
- Temporary group chats.
- Friend management.
- Business showcase.



# Interactive Map & Event Discovery



- Leaflet.js map with dynamic event pins.
- Filter events by date, location, and type.
- Pins change colour based on event status.
- Map doubles as a personalised social calendar.







# Business Model & Market Opportunity

- Free platform with optional donation.
- Sponsored pins for local businesses.
- Data insights for B2B monetisation.
- Google Calendar export integration.
- Social groups and families.
- Marketing managers & small businesses.
- University clubs and societies.
  - **TAM:** 938M
  - **SAM:** 375M
  - **TM:** 93.75M



# Financial Plan

## Year 1

OVERVIEW	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEPT	OCT	NOV	DEC	
CUMMULATIVE EXPENSES	€ 4,246.97	€ 4,493.94	€ 5,419.15	€ 11,844.36	€ 24,219.57	€ 38,405.78	€ 51,050.97	€ 62,803.66	€ 76,921.82	€ 93,317.68	€ 106,740.04	€ 121,199.40	
CUMMULATIVE INCOME	€ 23,000.00	€ 48,000.00	€ 48,000.00	€ 63,000.00	€ 63,000.00	€ 68,700.00	€ 120,010.00	€ 121,570.00	€ 124,130.00	€ 127,680.00	€ 131,780.00	€ 137,780.00	
STATUS	€ 18,753.03	€ 43,506.06	€ 42,580.85	€ 51,155.64	€ 38,780.43	€ 30,294.22	€ 68,959.03	€ 58,766.34	€ 47,208.18	€ 34,362.32	€ 25,039.96	€ 16,580.60	€ 16,580.60

## Year 2




OVERVIEW	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEPT	OCT	NOV	DEC	
CUMMULATIVE EXPENSES	€ 12,158.56	€ 24,547.12	€ 37,350.68	€ 56,234.24	€ 70,157.80	€ 84,481.36	€ 98,249.92	€ 111,788.48	€ 127,511.64	€ 141,629.80	€ 156,172.16	€ 172,704.52	
CUMMULATIVE INCOME	€ 20,620.60	€ 25,660.60	€ 31,200.60	€ 57,740.60	€ 67,280.60	€ 76,820.60	€ 87,860.60	€ 97,900.60	€ 113,960.60	€ 126,520.60	€ 139,620.60	€ 195,720.60	
STATUS	€ 8,462.04	€ 1,113.48	€ (6,150.08)	€ 1,506.36	€ (2,877.20)	€ (7,660.76)	€ (10,389.32)	€ (13,887.88)	€ (13,551.04)	€ (15,109.20)	€ (16,551.56)	€ 23,016.08	€ 23,016.08

## Year 3

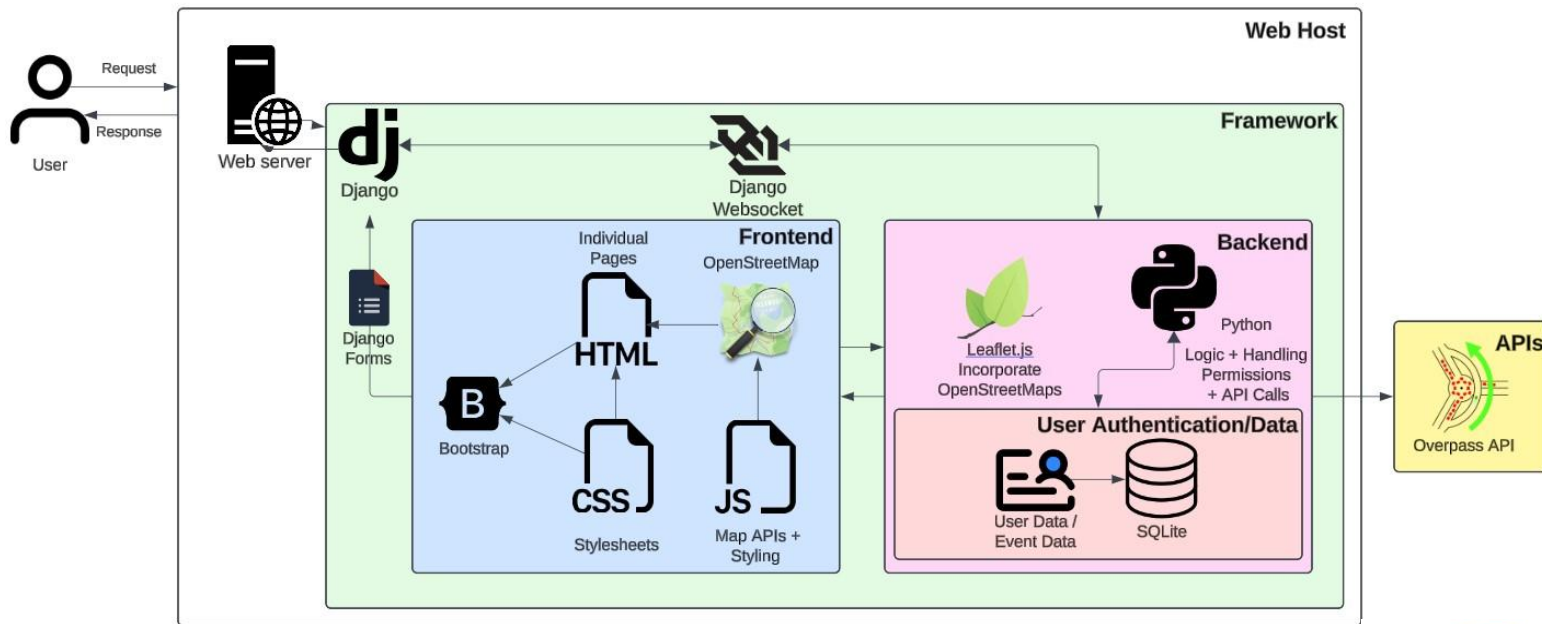
OVERVIEW	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEPT	OCT	NOV	DEC	
CUMMULATIVE EXPENSES	€ 15,669.36	€ 31,338.72	€ 48,958.08	€ 69,742.44	€ 86,256.80	€ 103,836.16	€ 119,965.52	€ 136,094.88	€ 154,134.24	€ 170,723.60	€ 187,382.96	€ 207,232.32	
CUMMULATIVE INCOME	€ 36,016.08	€ 49,016.08	€ 67,016.08	€ 80,516.08	€ 95,016.08	€ 115,016.08	€ 130,016.08	€ 145,016.08	€ 167,016.08	€ 184,016.08	€ 200,016.08	€ 224,016.08	
STATUS	€ 20,346.72	€ 17,677.36	€ 18,058.00	€ 10,773.64	€ 8,759.28	€ 11,179.92	€ 10,050.56	€ 8,921.20	€ 12,881.84	€ 13,292.48	€ 12,633.12	€ 16,783.76	€ 16,783.76



# Competitive Advantage

	Strengths	Weaknesses	Compared to PinPoint
	<ul style="list-style-type: none"><li>- Focus on Authentic Experiences.</li><li>- Collaborate with Friends when together.</li><li>- Daily Notifications.</li><li>- Interactive UI.</li></ul>	<ul style="list-style-type: none"><li>- Limited Feature Set.</li><li>- Lack of Communication.</li><li>- Low Engagement.</li></ul>	<ul style="list-style-type: none"><li>- BeReal captures spontaneous daily moments but lacks planning tools.</li><li>- PinPoint maintains authenticity while enabling event creation and real-world coordination.</li></ul>
	<ul style="list-style-type: none"><li>- Enormous User Base.</li><li>- Calendar Syncing</li><li>- Reminders through notifications.</li><li>- Strong Promotional Tools.</li></ul>	<ul style="list-style-type: none"><li>- Too large.</li><li>- Lower trust among younger audiences.</li><li>- More Formal Events as opposed to one-off hangouts.</li></ul>	<ul style="list-style-type: none"><li>- Facebook is cluttered and impersonal, catering to broad audiences.</li><li>- PinPoint offers focused, event planning within close social circles.</li></ul>
 Google Maps	<ul style="list-style-type: none"><li>- Detailed and reliable mapping.</li><li>- High trust in location accuracy and business reviews.</li><li>- Integrated with search and navigation.</li></ul>	<ul style="list-style-type: none"><li>- Not designed for social planning.</li><li>- Lacks group coordination features.</li><li>- Focuses more on business discovery than social experiences.</li></ul>	<ul style="list-style-type: none"><li>- Google Maps helps find places but lacks social planning tools.</li><li>- PinPoint combines discovery with real-time collaboration and event-specific chat.</li></ul>

# System Architecture





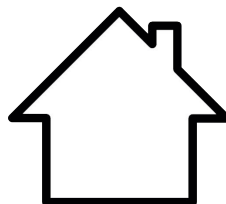
# Database Design & Models

- SQLite used for development.
- Key models: User, Event, RSVP, Message, FriendRequest.
- Relationships use Django's ORM and foreign keys.
- Chat messages and events tied to specific users and pins.



# UI/UX Design

- Clean, familiar interface using Bootstrap.
- Dark theme with accessibility contrast.
- Consistent iconography and navigation bar.





# Source Code Highlights

# Map Rendering

```
function initializeMap(lat, lon) {
  map = L.map('map').setView([lat, lon], 13);

  L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
    maxZoom: 19,
    attribution: '©copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a>'
  }).addTo(map);

  // Save user's location globally
  userLat = lat;
  userLon = lon;

  // User location marker and accuracy circle - Moved from out of function to inner
  userMarker = L.circleMarker([userLat, userLon], {
    radius: 8,
    color: '#3388ff',
    fillColor: '#3388ff',
    fillOpacity: 1
  }).addTo(map);

  accuracyCircle = L.circle([userLat, userLon], {
    radius: 1500,
    color: '#3388ff',
    fillColor: '#3388ff',
    fillOpacity: 0.2
  }).addTo(map);

  // Add event pins
  const today = new Date();
  events.forEach(event => {
    const markerIcon = new L.Icon({
      iconUrl: event.in_showcase
        ? '{% static "icons/showcase_pin.png" %}'
        : event.ongoing
          ? '{% static "icons/ongoing_pin.png" %}'
          : '{% static "icons/pin.png" %}',
      iconSize: [32, 32],
      iconAnchor: [16, 32],
      popupAnchor: [0, -32]
    });

    const marker = L.marker([event.lat, event.lng], { icon: markerIcon }).addTo(map);
```

- Initializes the map .
  - User's current location (lat, lon).
- Loads OpenStreetMap.
- Stores coordinates for later use.
- Adds a circle marker for the user's nearby location.
  - 1.5km radius accuracy circle.
- Iterates through event data.
- Uses different icons to represent event types.



# Event Creation (Public)

```

@login_required
def public_event_create(request):
    if request.method == "POST":
        form = EventForm(request.POST)

        if form.is_valid():
            event = form.save(commit=False)
            event.created_by = request.user
            event.is_public = True

            # Handle tags
            try:
                import json
                tag_data = request.POST.get('tags', '[]')
                event.tags = json.loads(tag_data)
            except:
                event.tags = []

            event.in_showcase = request.POST.get('in_showcase') == 'True'
            event.save()
            form.save_m2m()
            return redirect('event_detail', event_id=event.id)
        else:
            latitude = request.GET.get('latitude', '')
            longitude = request.GET.get('longitude', '')
            place_name = request.GET.get('name', '')
            form = EventForm(initial={
                'latitude': latitude,
                'longitude': longitude,
                'location_name': place_name,
                'name': place_name,
                'is_public': True,
                'tags': '[]'
            })
            form.fields['invitees'] = request.user.friends.none()

    return render(request, 'public_event_create.html', {'form': form})

```

- Authenticated users can create public events through this view.
- Sets is\_public = True and links the event to the logged-in user.
- Optionally checks if the event should appear in the showcase.
- Pre-fills data from the query parameters.
- Hides the invitees field (since public events don't target individuals).

# RSVP Handling

```

{% if user != event.created_by %}
  <form method="POST" action="{% url 'rsvp_event' event.id %}" class="mt-4">
    {% csrf_token %}
    <div class="input-group">
      <select name="status" class="form-select">
        <option value="yes">Yes</option>
        <option value="no">No</option>
        <option value="maybe">Maybe</option>
      </select>
      <button type="submit" class="btn btn-primary">RSVP</button>
    </div>
  </form>
{% endif %}

<div class="card p-3 bg-secondary mt-4">
  <h5 class="mb-3">RSVP Responses</h5>
  <p class="mb-2"><strong class="text-success">Yes:</strong>
    {% for rsvp in all_rsups %}
      {% if rsvp.status == "yes" %} {{ rsvp.user.username }}{% if not forloop.last %},
    {% endif %}{% endif %}
    {% endfor %}
  </p>
  <p class="mb-2"><strong class="text-danger">No:</strong>
    {% for rsvp in all_rsups %}
      {% if rsvp.status == "no" %} {{ rsvp.user.username }}{% if not forloop.last %},{%
    endif %}{% endif %}
    {% endfor %}
  </p>
  <p><strong class="text-warning">Tentative:</strong>
    {% for rsvp in all_rsups %}
      {% if rsvp.status == "maybe" %} {{ rsvp.user.username }}{% if not forloop.last
    %},{% endif %}{% endif %}
    {% endfor %}
  </p>
</div>

```

- Displays an RSVP form.
  - Users not hosting the event.
- Allows selection of attendance status:
  - Yes, No, Tentative.
- Submits status via POST.
- Below the form, responses are shown in a styled card.
- Categorizes and lists attendees.
- Clear view of attendee interest and availability.

# Real-Time Chat (Django Channels)



```
class ChatConsumer(AsyncWebsocketConsumer):
    async def connect(self):
        self.event_id = self.scope['url_route']['kwargs']['event_id']
        self.room_group_name = f"chat_{self.event_id}"

        await self.channel_layer.group_add(
            self.room_group_name,
            self.channel_name
        )
        await self.accept()

        messages = await self.get_chat_history(self.event_id)
        await self.send(text_data=json.dumps({
            "type": "chat_history",
            "messages": messages
        }))

    async def disconnect(self, close_code):
        await self.channel_layer.group_discard(
            self.room_group_name,
            self.channel_name
        )

    async def receive(self, text_data):
        data = json.loads(text_data)
        user = self.scope["user"]

        if not user.is_authenticated:
            await self.close()
            return

        if 'message' in data and data['message']:
            message = data['message']
            await self.save_message(user, message, self.event_id)
            await self.channel_layer.group_send(
                self.room_group_name,
                {
                    'type': 'chat_message',
                    'message': message,
                    'username': user.username
                }
            )
```

- WebSocket connections.
- Group tied to each event ID.
- Chat history upon connection.
- Processes incoming messages.
- Broadcasts messages in real-time.
- Removing users from the group on disconnect.



# Scalability & Future Plans

- Move to PostgreSQL + PostGIS.
- Switch to Google Maps API.
- API rate limiting & caching strategies.
- Containerisation with Docker for cloud deployment.



# Challenges Faced

- Real-time chat sync across devices.
- Timezone handling in event expiry.
- Group chat structure for public events.
- Chatrooms for concert attendees.



# Conclusion & Final Thoughts

- PinPoint solves a real, everyday problem.
- Combines multiple tools into one streamlined app.
- Scalable, with strong future potential.
- Built from the ground up with real users in mind.

# Thank You for your time!

Feel free to ask us any questions!

