

BandWay

Ido Bitton

Tal Yamin



BandWay
Where Music Meets Vacation

CONTENTS

Overview	4
The story behind Bandway.....	4
Problem	4
Solution	5
The Product	6
Design	7
Code Documentation.....	7
Server-Side Documentation	7
Key Components.....	8
Example Code Snippets	10
Client-Side Documentation	11
Key Components.....	11
Example Code Snippets	14
User Flow Chart	16
Technologies	16
The Benefits	18
The Innovation	18
Competitors	19
Marketing	19
Business Model.....	20
Revenue Streams:	20
Customer Segments:	20
Finances	21
Financing	21
Yield / Forecast	21
Required Financing	21
Marketing Strategy.....	21
Market Research.....	21
Brand Positioning.....	21
Customer Acquisition	21
Promotional Activities	22
Retention Strategies	22

Limitations.....	22
Low Resolution Hotel Image	22
Division of Labor	22
About Us	23
Pictures	23
Home Page	23
Services Package Finder Page	24
Package Dialog	24
Subscribe Page	25
Swagger API	26
Contact us	26

OVERVIEW

BandWay is a revolutionary concert vacation planning system designed to provide users with an unparalleled experience in organizing and enjoying their favorite music events. Leveraging cutting-edge technology and intelligent algorithms, BandWay offers personalized recommendations, seamless booking processes, and comprehensive trip management features.

THE STORY BEHIND BANDWAY

The story of BandWay's creation begins with a developer's honeymoon journey. As he and his wife, both passionate music lovers, embarked on planning their dream honeymoon, they envisioned a trip filled with concerts and live performances from their favorite bands. However, as they searched tirelessly for a solution that would streamline the process of planning their concert-filled itinerary, they were met with frustration. Existing platforms lacked the ability to efficiently discover concerts, book tickets, and organize travel arrangements all in one place.

PROBLEM

Planning a concert vacation can be a complex and fragmented process, involving multiple platforms for discovering concerts, booking tickets, arranging accommodations, and organizing travel. This disjointed approach is time-consuming, frustrating, and prone to errors.

First, finding the right concerts or festivals that align with one's musical tastes and schedule requires extensive searching across various websites and apps. Once a suitable event is identified, purchasing tickets often means navigating different ticketing platforms, each with its own procedures, fees, and availability issues.

Next, arranging accommodations and travel adds another layer of complexity. Users must search for hotels or other lodging options on separate booking sites, compare prices, and ensure the location is convenient for the event venue. Additionally, booking flights, rental cars, or other transportation involves visiting multiple travel booking websites, further complicating the process.

Coordinating all these elements—concert tickets, accommodation, and travel—into a cohesive itinerary is challenging. Users need to manage different booking references, and manually compile a schedule that integrates all aspects of the trip.

Moreover, the lack of integrated support and personalized recommendations means users miss out on potential experiences and conveniences. They must rely on general travel guides and reviews, which may not cater specifically to concert-goers, leaving gaps in their planning.

This fragmented approach not only consumes significant time and effort but also increases the likelihood of errors and missed opportunities, leading to a less enjoyable and more stressful concert vacation experience.

SOLUTION

BandWay offers an integrated platform that streamlines the entire concert vacation planning process. It combines concert discovery, ticket booking, accommodation arrangements, and travel planning into one cohesive system. Leveraging advanced technologies and cloud computing for scalability, BandWay provides a seamless booking experience. Users receive tailored suggestions for concerts and travel options, can book everything in one place, and manage their itineraries effortlessly. With features like social integration and email alerts for new concerts and deals, BandWay ensures a hassle-free and enjoyable planning experience, focusing on enhancing user satisfaction and convenience.

Integrating with major ticketing platforms, BandWay simplifies the planning of concert vacation. At the first step, allowing users to search for tickets directly through the app. This integration ensures a smooth and unified booking experience, where users can compare ticket prices and choose their preferred seating options.

Accommodation arrangements are another critical component of the concert vacation planning process, and BandWay excels in this area by partnering with leading hotel and vacation rental services. Users can easily browse a wide range of lodging options that fit their budget, preferences, and proximity to the concert venue. With detailed descriptions, photos, and user reviews, making informed decisions becomes straightforward.

Travel planning is also made effortless with BandWay, as it provides comprehensive tools for booking flights and car rentals. Users can arrange all their travel needs directly through the platform, benefiting from real-time comparisons of prices and schedules. BandWay ensures that all travel arrangements are synchronized with concert dates and times, providing users with the peace of mind that every logistical aspect of their trip is covered.

To keep users engaged and informed, BandWay offers email and push notifications for new concert announcements, ticket sales, and exclusive deals. Personalized alerts ensure that users

never miss out on opportunities that match their interests, further enhancing the overall experience.

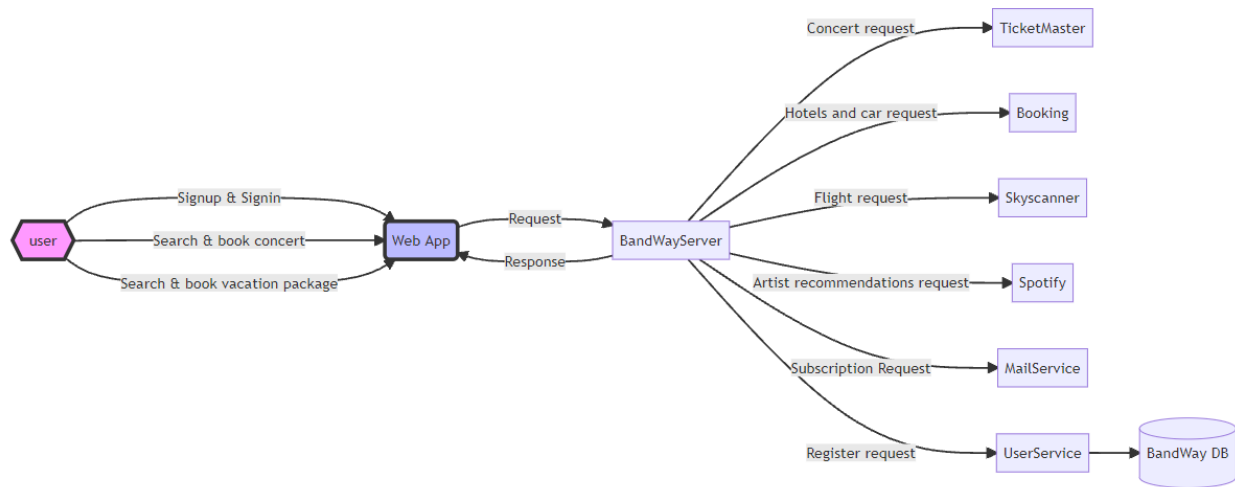
By integrating all these components into one unified platform, BandWay transforms the concert vacation planning process into a streamlined, enjoyable experience. Users save time and avoid the frustration of dealing with multiple disconnected services, allowing them to focus on what matters most: enjoying their favorite music events. BandWay's commitment to enhancing user satisfaction and convenience sets it apart as a leader in the concert vacation planning industry.

THE PRODUCT

Using BandWay is easy:

- Users can create an account or log in to BandWay using their credentials.
- Users can explore upcoming concerts, festivals, and events from a wide range of artists and genres.
- Users select a concert they're interested in attending and view detailed information such as date, time, venue, and ticket availability.
- Users seamlessly book tickets for the selected concert through the BandWay platform.
- Book Ground Service enables users to seamlessly arrange transportation, accommodations, and flights for their concert vacation, all within the BandWay platform.
- Users attend the concert and enjoy the music, knowing that BandWay has helped simplify their planning process.

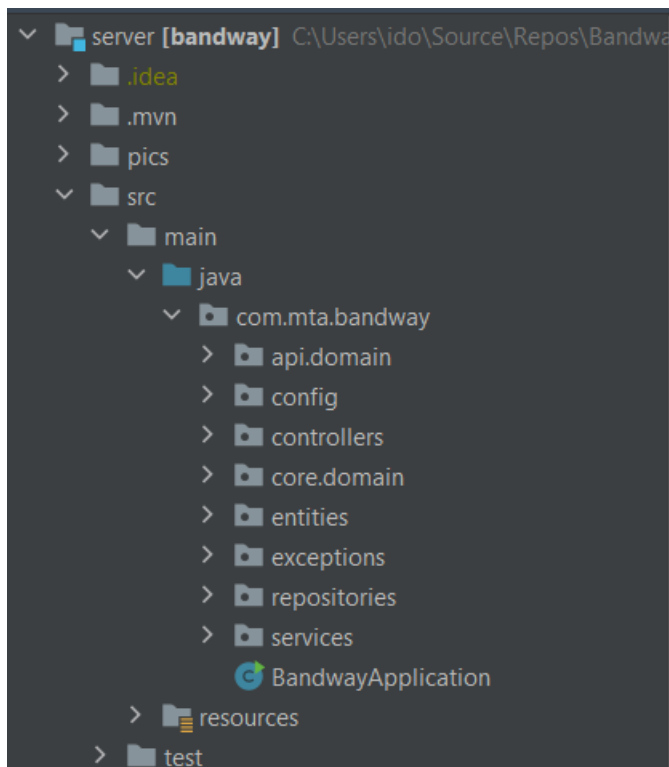
DESIGN



CODE DOCUMENTATION

The system is built using a client-server architecture, with the client side developed in TypeScript using React Redux, and the server side implemented in Java using the Spring framework.

SERVER-SIDE DOCUMENTATION



KEY COMPONENTS

BandwayApplication.java

- **Description:** The main entry point for the Spring Boot application.
- **Responsibilities:** Bootstraps the Spring Boot application.
- **Key Methods and Properties:**
 - `main()`: Starts the Spring application.

/api.domain

- **Description:** Define the communication interfaces.
- **Key Files and Functions:**
 - `FlightRequestDto.java`: Represents the request object for searching flights
 - `ConcertResponseDto.java`: Represents the response object for concert found

/core.domain

- **Description:** Define the core interfaces.
- **Key Files and Functions:**
 - `CarData.java`: Represents the car object for internal logic.

/controllers

- **Description:** Contains REST controllers that handle HTTP requests.
- **Key Files and Functions:**
 - `BandwayController.java`: Manages all endpoints of the server (e.g., login, registration, Serach Concert, etc.).

/services

- **Description:** Implements business logic and interactions with repositories.
- **Key Files and Functions:**
 - `UserService.java`: Contains methods for user-related operations.
 - `ConcertService.java`: Manages concert data operations.

/repositories

- **Description:** Interfaces for data access and interaction with the database.
- **Key Files and Functions:**
 - `UserRepository.java`: Provides methods to perform CRUD operations on user data.
 - `ConcertOrderRepository.java`: Handles database operations for concert data.
 - `FlightOrderRepository.java`: Handles database operations for flight data.

/entities

- **Description:** Defines entity classes representing database tables.
- **Key Files and Functions:**
 - `User.java`: Represents the user entity.
 - `Concert.java`: Represents the concert entity.

/exceptions

- **Description:** Defines exceptions that the system knows how to handle with.
- **Key Files and Functions:**
 - `UserAlreadyExistException.java`: A known issue of registration of existing user.

EXAMPLE CODE SNIPPETS

BandwayController.java

```
@RestController
@RequestMapping("/bandway")
@AllArgsConstructor
@CrossOrigin
@EnableScheduling
public class BandwayController {
    private final HotelService hotelService;
    private final ConcertService concertService;
    private final FlightService flightService;
    private final CarRentalService carRentalService;
    private final MailService mailService;
    private final UserService userService;

    // idobi
    @GetMapping(value = "/health", produces = "application/json", headers = "Accept=application/json")
    public ResponseEntity<String> health() { return ResponseEntity.ok( body: "Bandway is healthy!"); }

    // idobi +1
    @PostMapping(value = "/searchHotel", produces = "application/json", headers = "Accept=application/json")
    public ResponseEntity<List<HotelResponseDto>> searchHotel(@RequestBody HotelRequestDto requestHotelDto) {...}

    // idobi
    @GetMapping(value = "/searchConcert", produces = "application/json", headers = "Accept=application/json")
    public ResponseEntity<List<ConcertResponseDto>> searchConcert(@RequestParam String performer, @RequestParam(required = false) List<String> cities) {...}

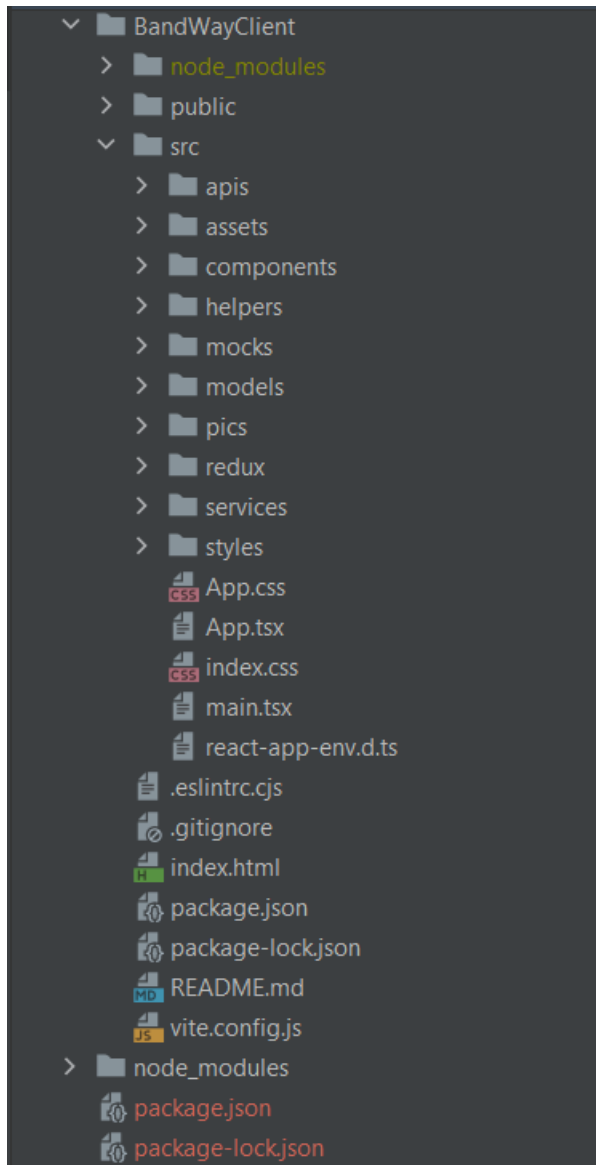
    // idobi +1
    @GetMapping(value = "/flightCityAutoComplete", produces = "application/json", headers = "Accept=application/json")
    public ResponseEntity<List<AutoCompleteCityResponseDto>> getCities(@RequestParam String text) {...}
```

UserService.java

```
public String registerUser(String firstName, String lastName, String username, String password, String email, String phone, Boolean isSubscribe) {
    Optional<User> findUser = userRepository.findByUsernameOrPhoneOrEmail(username, phone, email);
    if (findUser.isPresent()) {
        User user = findUser.get();
        if (user.getUsername().equals(username.toLowerCase()))
            throw new UserAlreadyExistsException("Username already exists");
        if (user.getPhone().equals(phone.toLowerCase()))
            throw new UserAlreadyExistsException("Phone already exists");
        if (user.getEmail().equals(email.toLowerCase()))
            throw new UserAlreadyExistsException("Email already exists");
    }
    User user = User.builder()
        .username(username.toLowerCase())
        .firstName(firstName.toLowerCase())
        .lastName(lastName.toLowerCase())
        .password(password)
        .email(email.toLowerCase())
        .phone(phone.toLowerCase())
        .isSubscribed(isSubscribe)
        .build();
    userRepository.save(user);
    return "User signed up successfully";
}

// usage idobi
public User loginUser(String username, String password) {
    Optional<User> findUser = userRepository.findByUsernameAndPassword(username, password);
    if (findUser.isPresent()) {
        return findUser.get();
    }
    throw new UserNotFoundException("User not found");
}
```

CLIENT-SIDE DOCUMENTATION



KEY COMPONENTS

App.tsx

- **Description:** The root component of the BandWay application.
- **Responsibilities:** Sets up the main layout and routing for the application.
- **Key Methods and Properties:**
 - `render()`: Renders the main structure of the application.

main.tsx

- **Description:** The entry point of the React application.
- **Responsibilities:** Renders the App component into the DOM.
- **Key Methods and Properties:**
 - `ReactDOM.render()`: Attaches the React app to the HTML DOM.

/redux

- **Description:** State management by Redux.
Contains Redux actions, reducers, and store configuration.
- **Key Files and Functions:**
 - `store.ts`: Configures the Redux store and middleware.
 - `actions.ts`: Defines actions related to concert data and package data.
 - `reducer.ts`: Manages the state related to concert data and package data.

/components

- **Description:** Manages React components and UI elements for the application. Each component file encapsulates specific UI elements and functionalities, promoting modularity and reusability within the React application architecture.
- **Key Files and Functions:**
 - `EventCard.ts`: Defines a component for displaying information about events, including details such as event name, date, and location.
 - `PackageCard.ts`: Implements a component for showcasing package details, such as accommodation options, flights and transportation choices.
 - `PackageDialog.ts`: Provides a dialog component for presenting detailed information about packages, including pricing, inclusions, and booking options.

/services

- **Description:** Manages business logic operations.
- **Key Files and Functions:**
 - `FlightService.ts`: Includes methods for handling flight type, duration and price.
 - `PackageBuilderService.ts`: Includes methods for packaging hotel, flight and car rental as unified object.

/apis

- **Description:** Handles API calls and external data fetching.
- **Key Files and Functions:**

Contains methods for making HTTP requests to the server.

- `EventApi.ts`
- `FlightApi.ts`
- `HotelApi.ts`

/models

- **Description:** Manages TypeScript objects representing entities and data structures used throughout the application, ensuring consistency and clarity in data handling.
- **Key Files and Functions:**
 - `Event.ts`: Defines the TypeScript object structure for representing event entities, including properties such as event name, date, location, and ticket information.
 - `Package.ts`: Specifies the TypeScript object structure for package entities, encompassing details such as package name, description, pricing, and inclusions.

EXAMPLE CODE SNIPPETS

EventApi.ts

```
import axios from 'axios';
import { EventResponse } from '../models/EventResponse';
import { Helpers } from '../helpers/helpers';
import { ArtistResponse } from '../models/ArtistResponse';

const helpers = new Helpers();

export class EventApi {

  BASE_URL: string = "http://localhost:8080/bandway";

  public async getEventsByPerformer(performer: string): Promise<EventResponse[]> {
    try {
      const queryPerformer: string = helpers.replaceSpacesWithUnderscores(performer);
      const response = await axios.get<EventResponse[]>(`${this.BASE_URL}/searchConcert?performer=${queryPerformer}`);
      return response.data;
    } catch (error) {
      throw new Error('Error fetching events by performer');
    }
  }

  public async getUpcomingEvents(): Promise<EventResponse[]> {
    try {
      const response = await axios.get<EventResponse[]>(`${this.BASE_URL}/upcomingConcert`);
      return response.data;
    } catch (error) {
      throw new Error('Error fetching upcoming events');
    }
  }

  public async getArtistAutoComplete(artistName:string): Promise<ArtistResponse[]> {
    try {
      const response = await axios.get<ArtistResponse[]>(`${this.BASE_URL}/artistAutoComplete?artistName=${artistName}`);
      return response.data;
    } catch (error) {
      throw new Error('Error fetching artists');
    }
  }
}
```

EventSearchResults.tsx

```
const EventSearchResults: React.FC = () => {
  const [events, setEvents] = useState<EventResponse[]>([]);
  const [isLoading, setIsLoading] = useState(true);
  const eventData = useSelector((state: AppState) => state.eventData);
  const navigate = useNavigate();

  // Load eventData from localStorage if available
  useEffect(() => {
    const storedEventData = localStorage.getItem('eventData');
    if (storedEventData) {
      const parsedEventData = JSON.parse(storedEventData);
      if (parsedEventData.performer) {
        const eventApi = new EventApi();
        eventApi.getEventsByPerformer(parsedEventData.performer)
          .then((data) => {
            if (parsedEventData.toCity) {
              const filteredEvents = data.filter((event) => event.city === parsedEventData.toCity);
              setEvents(filteredEvents);
            } else {
              setEvents(data); // Set all events if toCity is not provided
            }
            setIsLoading(false);
            console.log(events);
          })
          .catch((error) => {
            console.error('Error fetching event data:', error);
            setIsLoading(false);
            navigate('/error');
          });
      }
    } else {
      setIsLoading(false); // No eventData in localStorage, so stop loading
    }
  }, [navigate]);

  const handleClickOnSearchAgain = () => {
    navigate("/home");
  }

  const getPerformerNameFromLocalStorage = () => {
    const storedEventData = localStorage.getItem('eventData');
    if (storedEventData) {
      const parsedEventData = JSON.parse(storedEventData);
      return parsedEventData.performer;
    }
    return '';
  }
}
```

```
return (
  <>
    <CssBaseline />
    <TopContent mainText={events.length > 0 ? "We Discovered Outstanding Results for You..." : "We are Looking for Results for You..."} subText="" />
    <Box display="flex" justifyContent="center" >
      <div>
        <isloading ? (
          <Loader loadingMessage={`Loading ${getPerformerNameFromLocalStorage()} events...`} />
        ) : events.length > 0 ? (
          <UpcomingEvents events={events} title={` ${eventData?.performer} Events${eventData?.toCity ? ` in ${eventData?.toCity}` : ''}` />
        ) : (
          <Stack justifyContent={'center'} alignItems={'center'} sx={{ p: 8 }}>
            <Typography variant="h4" color="textSecondary" textAlign="center">Looks like there are no events matching your search</Typography>
            <Typography variant="h5" color="textSecondary">Consider altering either the performer or destination for better results.</Typography>
            <ActionButton variant="contained" onClick={handleClickOnSearchAgain} style={{ width: '350px', height: '80px' }}>Let's search again</ActionButton>
          </Stack>
        )
      </div>
    </Box>
    <Footer />
  </>
);

export default EventSearchResults;
```

USER FLOW CHART

You can find the flow chart in the next [link](#)

TECHNOLOGIES

- **Spring**

Spring is a powerful, feature-rich framework for building Java-based enterprise applications. It provides comprehensive infrastructure support for developing robust and scalable applications, simplifying the development of Java EE applications.

- **Java**

Java is a high-level, object-oriented programming language known for its portability, performance, and security features. It is widely used for building cross-platform applications, ranging from web and mobile apps to enterprise solutions.

- **Hibernate**

Hibernate is an object-relational mapping (ORM) tool for Java, which simplifies database interactions by mapping Java objects to database tables. It helps in managing database operations without writing extensive SQL code.

- **Typescript**

TypeScript is a statically typed superset of JavaScript that adds type safety and modern programming features to JavaScript. It enhances code quality and maintainability, making it easier to develop large-scale applications.

- **React**

React is a popular JavaScript library for building user interfaces, particularly for single-page applications. Developed by Facebook, it allows developers to create reusable UI components and manage the state of applications efficiently.

- **Material UI**

Material UI is a popular React UI framework that implements Google's Material Design principles. It provides a set of reusable and customizable UI components, making it easier to build visually appealing and consistent user interfaces.

- **Redux**

Redux is a state management library for JavaScript applications, often used with React to manage complex application state. It operates on principles such as a single source of truth, immutable state changes through dispatched actions, and pure functions for state transformations, making it ideal for applications with intricate state logic.

- **Axios**

Axios is a popular promise-based HTTP client for making asynchronous requests to servers, compatible with both browser and Node.js environments. It features interceptors for request/response handling, automatic JSON transformations, and simplified error handling, making it a preferred choice for integrating with RESTful APIs and managing data fetching in web applications.

- **Rapid API:**

RapidAPI is a platform for discovering, connecting to, and managing APIs. It simplifies API integration by providing a marketplace for developers to find and connect to thousands of public APIs and manage their API connections in one place.

- **Postgres:**

Postgres, or PostgreSQL, is a powerful, open-source object-relational database system. It is known for its robustness, extensibility, and standards compliance, making it suitable for a wide range of applications, from small projects to large-scale enterprise systems.

- **Docker:**

Docker is a platform for developing, shipping, and running applications inside lightweight, portable containers. It enables developers to package applications with all their dependencies, ensuring consistent environments across different stages of development and deployment.

- **GitHub:**

GitHub is a web-based platform for version control and collaboration using Git. It allows developers to host and review code, manage projects, and collaborate on software development with features like pull requests, issue tracking, and continuous integration.

THE BENEFITS

- **Integrated Platform:** Combines concert discovery, ticket booking, accommodation arrangements, and travel planning into one cohesive system.
- **Seamless Booking Experience:** Leverages advanced technologies and cloud computing for scalability, providing a smooth and efficient booking process.
- **Tailored Suggestions:** Offers personalized recommendations for concerts and travel options, enhancing the user experience.
- **Unified Ticket Booking:** Integrates with major ticketing platforms, allowing users to search for, compare, and book tickets through the app.
- **Comprehensive Accommodation Options:** Partners with leading hotel and vacation rental services, providing a wide range of lodging options with detailed descriptions, photos, and user rating.
- **Effortless Travel Planning:** Provides tools for booking flights and car rentals, with real-time comparisons of prices and schedules, ensuring all travel arrangements align with concert dates.
- **Personalized Notifications:** Sends email and push notifications for new concert announcements, ticket sales, and exclusive deals, ensuring users never miss out on opportunities.
- **Time and Frustration Saver:** Combines all necessary services into one platform, saving users time and avoiding the frustration of managing multiple disconnected services.
- **Enhanced User Satisfaction:** Focuses on user convenience and satisfaction, setting BandWay apart as a leader in the concert vacation planning industry.

THE INNOVATION

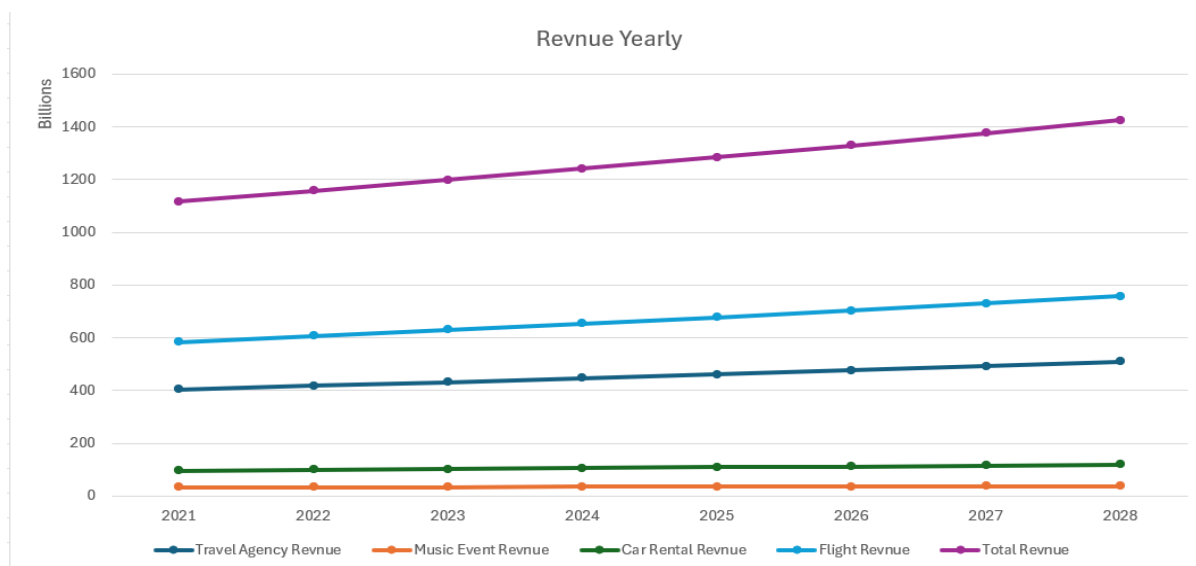
BandWay lies in its integration of personalized concert recommendations, seamless booking functionalities, and comprehensive trip management tools within a single platform. Unlike existing solutions that focus on either concert discovery, ticket booking, or travel planning separately, BandWay offers a holistic approach to concert vacation planning, providing users with an all-in-one solution that simplifies the entire process. This unique combination of features sets BandWay apart as a one-of-a-kind platform in the market, catering specifically to the needs of music enthusiasts who seek a streamlined and enjoyable planning experience for their concert vacations.

COMPETITORS

	Car rental	Book concert	Book hotel	Book flight	Find concert
Booking.com	✓	✗	✓	✓	✗
ticketmaster®	✗	✓	✗	✗	✓
songkick	✗	✗	✗	✗	✓
bandsintown	✗	✗	✗	✗	✓
eventbrite	✗	✗	✗	✗	✓
BandWay Where Music Meets Vacation	✓	✓	✓	✓	✓

MARKETING

- The Hotels booking market size is projected to grow by 15.6% (2022-2028) resulting in a market volume of **\$508.9** by 2028, from **\$446.50B** in 2021. [link](#)
- The Music Events market worldwide is projected to grow by 2.12% (2024-2028) resulting in a market volume of **\$37.22B** in 2028, from **\$34.23B** in 2023. [link](#)
- The Car Rentals market worldwide is projected to grow by 3.14% (2024-2028) resulting in a market volume of **\$116.00B** in 2028. from **\$102.50B** in 2023. [link](#)
- The Flight booking market worldwide is projected to grow by 3.75% (2024-2028) resulting in a market volume of **\$630B** in 2028. from **\$730B** in 2024. [link](#)



BUSINESS MODEL

REVENUE STREAMS:

1. **Commission Fees:** Earned from bookings made through BandWay for concerts, hotels, flights, and vacation packages.
2. **Subscription Plans:** Premium features and services offered through a subscription model.
3. **Advertising:** Revenue from targeted advertising for music events, travel services, and related products.
4. **Affiliate Marketing:** Partnerships with travel agencies, ticketing platforms, and other related services.
5. **Data Monetization:** Offering anonymized data insights to event organizers, travel agencies, and marketers.

CUSTOMER SEGMENTS:

1. **Music Enthusiasts:** Individuals who regularly attend concerts and music festivals. Travelers: People who combine travel with attending music events.
2. **Event Organizers:** Concert promoters and event managers looking for platforms to promote their events.
3. **Travel Agencies:** Agencies that offer travel packages and can benefit from integrating concert experiences.

FINANCES

FINANCING:

- Seed funding: \$500,000 to cover initial development, marketing, and operational costs.
- Series A: \$2 million to scale the platform, enhance features, and expand marketing efforts.
- Series B: \$5 million to optimize operations, introduce new services, and expand globally.

YIELD / FORECAST:

- Year 1: Revenue - \$1 million; Expenses - \$800,000; Profit - \$200,000.
- Year 2: Revenue - \$5 million; Expenses - \$3 million; Profit - \$2 million.
- Year 3: Revenue - \$15 million; Expenses - \$10 million; Profit - \$5 million.

REQUIRED FINANCING:

- Initial investment required to launch the platform and cover operational costs for the first year: \$1 million.
- Additional funding for scaling and global expansion in subsequent years: \$7 million over two years.

MARKETING STRATEGY

MARKET RESEARCH:

- Identify target demographics and their preferences.
- Analyze competitors and market trends.

BRAND POSITIONING:

- Position BandWay as the go-to platform for concert vacation planning.

CUSTOMER ACQUISITION:

- Leverage social media marketing to reach music enthusiasts.
- Utilize influencer marketing and partnerships with music blogs and websites.
- Implement SEO and content marketing strategies to attract organic traffic.

PROMOTIONAL ACTIVITIES:

- Offer limited-time discounts and promotions for early adopters.
- Create referral programs to incentivize existing users to invite friends.

RETENTION STRATEGIES:

- Implement a loyalty program for repeat customers.
- Send personalized recommendations and exclusive offers to subscribers.
- Provide excellent customer support to ensure a positive user experience.

LIMITATIONS

LOW RESOLUTION HOTEL IMAGE

- **Reference:**
<https://rapidapi.com/ntd119/api/booking-com18/discussions/137532>
- **Limitation:**
The Booking.com API currently provides images that are small and of low resolution.
- **Impact:**
When these images are scaled up, they appear blurry and of poor quality, which negatively affects the visual appeal and user experience in applications using these images.
- **Developer's Response:**
The issue is recognized by the developer.
A solution is being worked on, and users will be informed when the problem is resolved.

DIVISION OF LABOR

At BandWay, we have strategically divided our development efforts to leverage the strengths and expertise of our team members, ensuring the efficient and high-quality creation of our platform. Ido is responsible for developing the backend side of the application, utilizing Java and the Spring framework to build robust server-side functionalities, manage database interactions, and integrate with third-party services such as TicketMaster, Booking.com, Skyscanner, and Spotify. Meanwhile, Tal is focused on the client side, employing TypeScript with React Redux to create a dynamic and user-friendly interface. Tal ensures that our users have a seamless experience navigating the application, booking concerts and vacation packages, and receiving personalized recommendations. This division of labor allows us to develop a comprehensive and cohesive system that meets the needs of our users efficiently.

ABOUT US

- **Ido Bitton:**

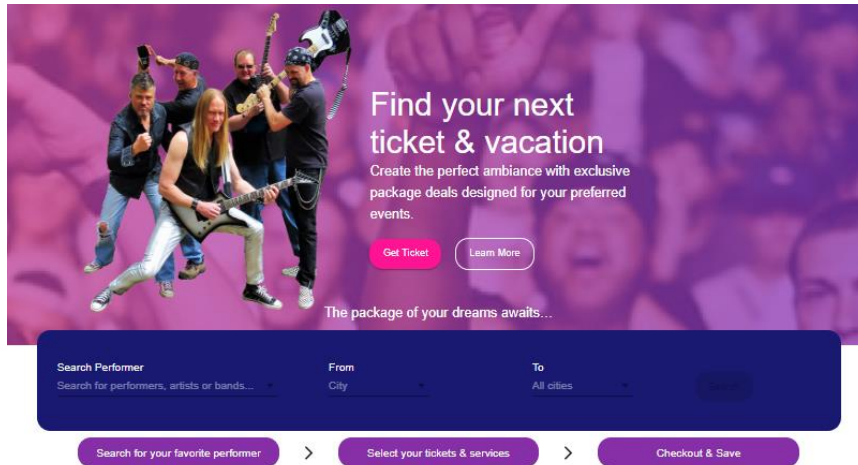
- I am a fourth-year computer science student living in Israel, with seven years of experience as a data engineer and backend developer. Beyond my professional life, I am happily married, dedicated father to one baby girl and a puppy dog owner .
- Linkdin: <https://il.linkedin.com/in/ido-bitton-b8a298163>

- **Tal Yamin:**

- I am a fourth-year computer science student, currently working as a QA Automation Engineer. On a personal note, I have been happily married for nearly three years, and last month, I became a proud father to a beautiful baby boy.
- Linkdin: <https://www.linkedin.com/in/tal-yamin-5a478a173/>

PICTURES

HOME PAGE



Upcoming Events



SERVICES PACKAGE FINDER PAGE

Services Package Finder

Explore our comprehensive service package finder for a complete and enhanced experience!

Services Date Range
Select Date Range

Occupancy
Select Occupancy

Services Budget
Select Budget


From
Select City

To
Select City

Search

Filters: HOTEL ✓ FLIGHT ✓ CAR RENTAL ✓

Best Deals



London
United Kingdom

25 April · 3 nights

Butlers Townhouse


★★★★

Rating: 7.7

✈ Flight is included

🚗 Car rental is included

Start from
\$2121
per person



London
United Kingdom

25 April · 3 nights

Sandymount Hotel


★★★★

Rating: 8.5

✈ Flight is included

🚗 Car rental is included

Start from
\$2122
per person



London
United Kingdom

25 April · 3 nights

Herbert Park Hotel

★★★★

Rating: 8.6

✈ Flight is included


🚗 Car rental is included

Start from
\$2306
per person

PACKAGE DIALOG

Your vacation details

Vacation to London
4/25/24 - 4/28/24
3 nights



Your Hotel:

Butlers Townhouse

★★★★

Rating: 7.7

adults: 1, children: 0

rooms: 1

Start from \$859 per person

[See hotel availability](#)

Choose Your Flight:

Outbound Flights: Direct Flight - 10:10 (TLV) - 13:35 (LHR) - 6hr 25m

Return Flights: Direct Flight - 09:45 (LTH) - 16:36 (TLV) - 4hr 50m

Start from \$1756.84 per person

Outbound Flights: Direct Flight - 16:20 (TLV) - 19:50 (LHR) - 6hr 30m

Return Flights: Connection Flight - 06:40 (LHR) - 15:05 (TLV) - 6hr 25m

Start from \$1475 per person

Outbound Flights: Direct Flight - 10:10 (TLV) - 13:35 (LHR) - 6hr 25m

Return Flights: Direct Flight - 22:10 (LHR) - 05:05 (TLV) - 4hr 55m

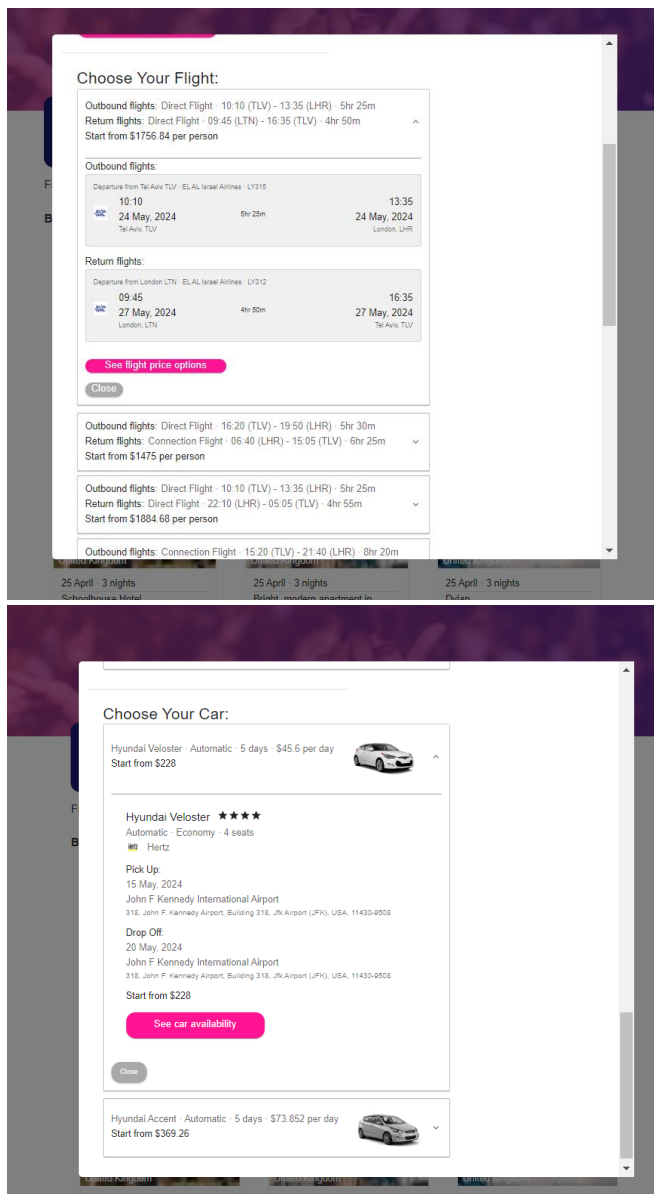
Start from \$1884.68 per person

25 April · 3 nights
Schoolhouse Hotel

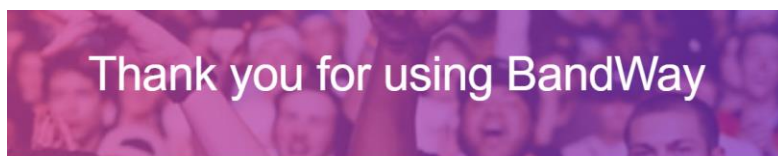
25 April · 3 nights
Bright modern apartment in

25 April · 3 nights
Dulux

24



SUBSCRIBE PAGE



Let's stay connected.

We recommend subscribing to stay updated with more events and vacation packages.

SWAGGER API

bandway-controller		^
POST	/bandway/unsubscribeMail	▼
POST	/bandway/subscribeMail	▼
POST	/bandway/sendMessageAllSubscribedUsers	▼
POST	/bandway/searchRoundWayFlight	▼
POST	/bandway/searchOneWayFlight	▼
POST	/bandway/searchHotel	▼
POST	/bandway/searchCarRental	▼
POST	/bandway/registerUser	▼
GET	/bandway/upcomingConcert	▼
GET	/bandway/searchConcert	▼
GET	/bandway/login	▼
GET	/bandway/health	▼
GET	/bandway/getHotelLink	▼
GET	/bandway/flightPrice	▼
GET	/bandway/flightCityAutoComplete	▼
GET	/bandway/carRentalCityAutoComplete	▼
GET	/bandway/artistSpotifyLink	▼
GET	/bandway/artistAutoComplete	▼

CONTACT US

Website: <https://bandway.com>

Mail: Bandway4@gmail.com