ALPINE EDGE: SKI MARKETPLACE PLATFORM

Project Report - Team 16-1

TEAM MEMBERS

David Poston	Github: dapo2410
Charlie Kasic	GitHub: Chaka5841
Agustin Garcia-Huidobro	GitHub: agustinghp

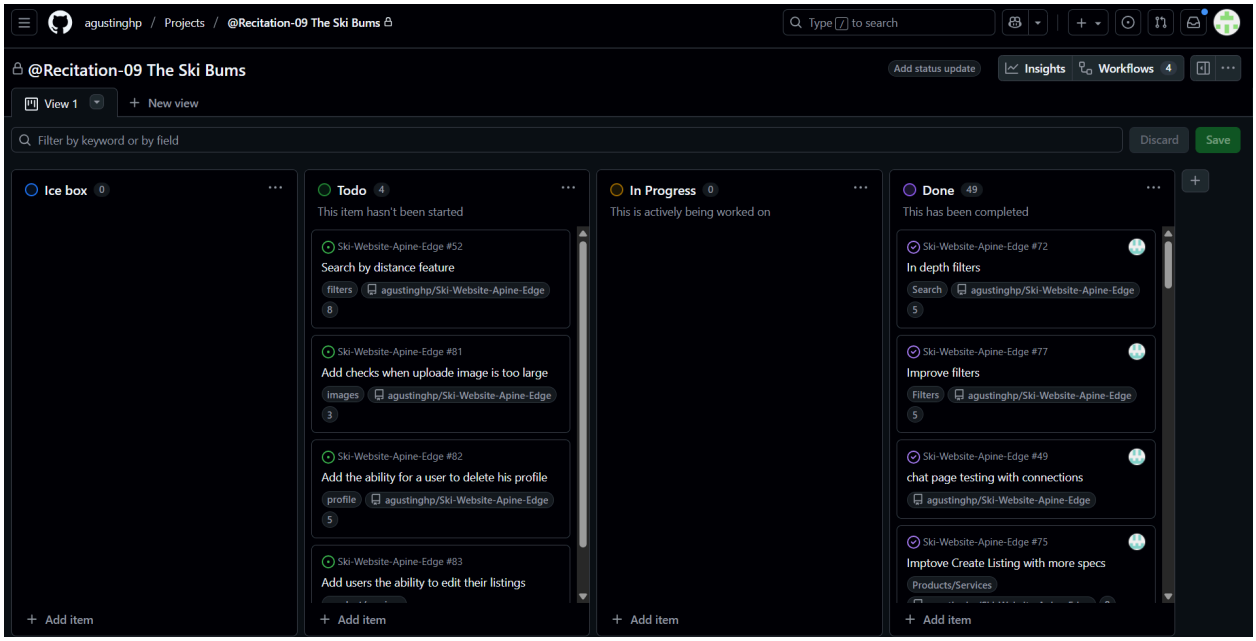
PROJECT DESCRIPTION

Alpine Edge is a comprehensive ski marketplace platform designed to connect ski equipment buyers, sellers, and service providers. The application enables users to list and search for ski products (equipment, gear) and services (ski tuning, lessons, rentals) in a centralized marketplace. Key features include user authentication, real-time messaging between buyers and sellers, advanced search with filtering capabilities, image upload for listings, location-based services, and a responsive design optimized for both desktop and mobile devices. The platform facilitates secure transactions and communications within the skiing community, making it easier for enthusiasts to find quality equipment and trusted service providers.

PROJECT TRACKER - GITHUB PROJECT BOARD

GitHub Repository Link:
https://github.com/agustinghp/Ski-Website-Apine-Edge
Project Board Link:
https://github.com/users/agustinghp/projects/1

Screenshot:



DEMO VIDEO

Video Link:
https://github.com/agustinghp/Ski-Website-Apine-Edge/blob/main/Milestone%20Submissions/Video%20Demo.mp4

The demo video showcases the complete user journey including registration, product/service listing creation, search functionality, messaging between users, and the overall user experience.

VERSION CONTROL SYSTEM (VCS)

Git Repository:
https://github.com/agustinghp/Ski-Website-Apine-Edge

INDIVIDUAL CONTRIBUTIONS

David Poston:

Implemented the image storage system for product listings and user profiles using Multer and Sharp libraries. Developed the complete product and service implementation including database schema design, CRUD operations, and integration with the front end. Created the product detail pages with dynamic rendering of images, specifications, and seller information. Worked on the file upload middleware and image optimization to ensure fast loading times.

Technologies Used:
• Node.js
• PostgreSQL
• Multer
• Sharp
• Express.js

Features Contributed:
• Image storage and upload system
• Product CRUD operations
• Service CRUD operations
• Product detail pages
• Database schema design

Charlie Kasic:

Developed the pages and partials for the handlebars HTML front end design. Used CSS to create a style and theme for the page. Implemented the javascript middleware for registration, login and path navigation. Integrated unique search by location features into search and advanced search pages with precise location and rounding. Integrated database and user interface. Fixed important bugs with database indexing for uploading images to posts properly.

Technologies Used:

• HTML/Bootstrap/Handlebars
• Javascript authentication middleware.
• SQL indexing

• Haversine formula for distance in JS
• Google API integration with search

Features Contributed:
• Front end design pages and partials
• Login/registration middleware
• Authentication checking
• Distance filter/ search by radius

Agustin Garcia-Huidobro:

Implemented the real-time chat messaging system using Socket.io for bidirectional communication between connected users, saving the chat messages using PostgreSQL Developed the advanced search functionality with filtering by product type, price range, brand, and category-specific attributes (dimensions, sizes, etc.), enabling users to search products, services, and users with multiple filter combinations. Enhanced the product listing page, with support for a variety of products (skis, snowboards, helmets, boots, poles, goggles, gloves, jackets, pants and other), with form fields that update based on the selected category. Also added robust error handling for the page for things like image size being too large or bound checking for numerical inputs and more. Integrated Google Maps Places API for the location autocomplete on the registration page, which allows users to select their location with automatic coordinate extraction, while making sure other users can only see other people's cities but not exact coordinates. Built and implemented the connections feature with the database, and the html pages using the handlebars. Users can send requests, reject them, or send them again after they reject someone. A connection unlocks the ability to chat with the connected user.

Technologies Used:
• Node.js
• PostgreSQL
• Socket.io
• Express.js
• JavaScript
• Handlebars
• Google Maps API

Features Contributed:
• Real-time chat messaging system
• Advanced search with multi-filter functionality
• Product listing form with dynamic fields for multiple product types
• Location autocomplete for user registration

• Connections/friend request system
• Database schema for products, services, users, connections and messages
• Other users profile page (Not your own but when you are viewing someone else's profile)

USE CASE DIAGRAM

The following use case diagram illustrates the interactions between different actors (Buyers, Sellers, Service Providers, Guests) and the system functionalities.

ACTORS:

• Guest User: Can browse listings, view product/service details, search and filter

• Registered User: All guest capabilities plus create listings, message other users, manage profile

• Seller: List products, manage inventory, respond to inquiries, update listings

• Service Provider: List services, manage bookings, communicate with clients

MAIN USE CASES:

1. User Registration & Authentication
   - Login
   - Register
   - Logout

2. Browse & Search Products/Services
   - Search by keywords
   - Filter by price, length, width, location
   - View search results
   - Sort results

3. Create Listings
   - Add product with specifications
   - Add service offering
   - Upload images
   - Set pricing

4. Manage Listings
   - Delete listings
   - View own listings
   - View other listings

5. Real-time Messaging
   - Send messages to other users
   - Receive messages instantly
   - View chat history

6. User Profile Management
   - Update location
   - View own listings
   - Manage account information

7. View Detailed Listings
   - View product specifications
   - View seller information
   - Contact seller/provider options

WIREFRAMES

The following wireframes were created during the design phase of the project, **ALL WIREFRAMES IN GITHUB REPO**:

1. Home Page
   Landing page with featured listings, navigation menu, and call-to-action buttons for registration.

2. Login/Registration Pages
   User authentication forms with input validation and error handling. Inputs: Username, Location input box with google autocomplete, email and password

3. Search Page
   Advanced search interface with filters (type and location) and sort Options.

4. Advanced Search Page
Page with more advanced search features to specify different products and their dimensions, can filter by brands, size and width/length for skis and snowboards.

5. Product/Service Detail Page
Comprehensive view of listing details, images, seller information, and contact options.

6. Create Listing Page
Form interface for adding new products or services with image upload capability.

7. User Profile Page
Dashboard showing user's own listings, account information, and management Options, such as changing profile image.

8. Chat/Messaging Page
Real-time messaging interface with conversation history and contact list.

9. Connections Page
List of all current, pending and incoming connection requests.

10. Other users Profile Page
Dashboard showing this user’s listing, connection status and information if available.

USER ACCEPTANCE TEST RESULTS

Based on the User Acceptance Test Plan from Lab 10, the following tests were Conducted.
Test cases:

1. User registration and login
2. Product Posting
3. Chat testing

KEY OBSERVATIONS FROM USER TESTING:

For all the testing, the application worked as expected:
They were able to create a user, and it automatically logged them in their account
They were able to post a ski product, and it appeared in the search results.

They were also able to chat with other users in real time.

Upon further testing we received feedback that our search page was lacking some clarity and the ability to do more filtering. So we added the advanced search page with a lot more filtering and better styled the regular search page.

DEPLOYMENT

Live Application URL:
https://alpineedge-web.onrender.com/

TECHNOLOGY STACK:
Backend:
• Node.js with Express.js framework

Database:
• PostgreSQL 15

Template Engine:
• Handlebars.js

Real-time Communication:
• Socket.io

Image Processing:
• Multer + Sharp

Frontend:
• Bootstrap 5, CSS3, JavaScript ES6+

Containerization:
• Docker & Docker Compose

Session Management:
• Express-session with connect-pg-simple