**Snowboard Community Project**

**1.** **Application Purpose**

The purpose of this project is to create a snowboard community where riders can interact with friends and other members of the community, as well as check snow reports and stay up to date with their favorite resorts.

**2.** **Functional Application Requirements:**

The functional requirements for the backend of this project are listed below.

CRUD+1 operations should be support for all of the entities and support admin and user roles:

* Using external basic weather api, display 7 day foracst for the desired resort
* User should be able to save favorite resorts and see this info in their dashboard
* User should be able to add friends for easier interaction
* Keep track of where your friends are riding
* Be able to see posts friends make in the forum
* Users should be able to add, edit, and delete their own forum post
* Only able to add post if they have an account registered
* Admin should have the ability to add, edit, and delete any post in the forum
* User dashboard should display favorite resorts, friends list, and top post made by the logged in user as well as their friends
* Possibly add in some sort of mapping to be able to see where friends are
* Possible check in feature

2a. Design

* *Planned frontend layout if applicable, use case flows between different features/screens.* ***For backend applications, a documented list of expected endpoints*** *with a brief description of function they serve, relationship(s) to any specific frontend features.*

The Model schemas as presently identified:

* USER MODEL
* User ID
* User Email
* User Name
* List of friends
* List of favorite resorts
* List of forum posts
* Resort Model
* Resort ID
* Resort Name
* Resort Location
* Resort Elevation
* Current weather from api
* <https://openweathermap.org/api/one-call-api>
* Forum MODEL
* List of Posts
* Forum Post Model
* ID
* User ID
* Category
* Post

Enpoints

* Add/edit/delete post
* User should be able to add/edit/and delete only their posts
* Add/Edit/And delete resort (Admin)
* Add Friend (user)
* Add resort to favorites (user)

2b. Test Cases

* ***Each requirement should have at least one test case****. Both for frontend and backend (via Postman)   
    
  Students must confirm that each test case passes prior to marking a requirement complete. Use Issue tickets, project board, and any other methods you choose to track progress!*

**3. Non-Functional Application Requirements**

3a. Assumptions About the Application

* *Note any assumptions the user should have about this application and its features (e.g. “this is only the first phase of development and does not represent the final version of this feature.”).*

3b. Application Design Considerations

* *Put some time into thinking about the overall design of your application/solution and note things that should be considered that may impact how you design your application. (e.g. will require integration into some sort of payment processing service to handle transactions.)*

**4.** **Future Enhancements**

*This section is used to capture ideas for future new enhancements and/or areas of the application that may need additional work, including documentation updates, code that needs refactoring due to performance, stability, or other concerns.*

* incorpate a more complex api that displays the snow conditions and detail for each resort
* api i found only has a 60 day free trial but has the the information I am trying to get
* in the future, I would like to use this api instead of the basic seven day forecast one I will use to create the initial version of this app
* <https://www.worldweatheronline.com/developer/api/docs/ski-weather-api.aspx#qparameter>