## Project Name

Atlas Itinerary

## Team Member Names

Gabriel Taylor, Dustin Zook, Sorelle Djuissi, Gabriyel Sorensen

## Abstract

Atlas Itinerary is a personal travel planning web app that lets users design detailed time-slotted trips without the pressure of booking. After creating an account, users can plan multiple trips, search destinations, and add lodging, restaurants, attractions, and activities. A defining characteristic is the timeline where every item can be placed on a specific day and time, so the schedule is easy to see and follow.

The app focuses on straightforward trip management, allowing users to create, edit, and save itineraries, assign lodging to nights, and meals to time slots, and drag items into a daily schedule. The goal is to simplify the planning by keeping choices and timing in one place. When it's time to book, the plan is already organized. The web app also works well on phones for quick updates while traveling.

o

Do not simply copy the abstract from your proposal. This should be a refined

abstract with a clearer vision.

4.

Tools & Technologies

## Initial Set of Technologies

* Platform: Web. Atlas Itinerary is a web-based program. Most of our group knows web-based program functionality, so by making one, we are more prepared for the task.
* Browser: Firefox. Firefox is the search engine on which we will make our web-based program. Our group is more familiar with programming web applications on Firefox.
* Operating System: Windows. All of our members have Windows operating systems by default. Instead of trying to learn how to make the application on Apple through a VM, we have elected to use Windows.
* IDE: IntelliJ. IntelliJ is the IDE our group feels most comfortable programming in. So instead of stretching to learn another IDE we elected to use this one. Preferred IDE for developing in Java.
* Languages: Java. For developing backend services for our Web application.
* HTML, CSS, JavaScript. Necessary for designing web pages and understanding how users will interact with the project..
* Server Software: TBD
* Communication Software: Discord. Allows for communication outside of class. Non-professional contact method to encourage bonding and familiarity among members.
* AI: not to use for coding, but to ask basic questions.

## Requirements list

1. Sign-in Form
   1. Upon accessing the domain, users will be greeted with a login screen
   2. Two boxes, one for username, one for password
   3. Login button and signup button
   4. If login is not recognized, a message appears below the username and password box letting users know it is not recognized.
   5. Need help option for recovering the user name and/or password
      1. Two recovery options will be provided: one for retrieving the username and another to reset the password
   6. In itinerary modification mode, button to save the itinerary to the account.
   7. Option to sign in using a Google account
2. Sign-Up Form
   1. The signup page offers two boxes, one for username and one for password.
   2. If the username is already in use, below the username box, the user will be told the username already exists
   3. Passwords will be required to be between 8 and 16 characters with at least one uppercase letter, lowercase letter, special character, and number.
   4. On the signup page an option to sign up through the user's Google account.
3. Home Page will be rendered once a user logs in
   1. About Section
4. Navigation Bar
   1. Create Itineraries
      1. Once clicked, the user will be presented a page where they can create itineraries
      2. There will be a label for their Travel Destination
      3. There will be an input box for their Travel Destination
      4. The application will be able to detect the user’s current location and generate suggestions for their Travel Destination
         1. Once the destination searched for is found, the user will be presented a sidebar with several categories
            1. Lodging is the Default category selected

Hotel

Bed and breakfast

Specialty lodging

* + - * 1. Dining

Can be sorted by Subcategories

* + - * 1. Attractions

Subcatagories

Filters for attractions include

* + - * 1. A drop-down menu will be above where locations are displayed containing a variety of sub-categories to filter locations
        2. Once a category is selected, locations of the category will be displayed as a list of cards

Cards will consist of an image of the location and the name of the location

* + - * 1. There will be a paging system so that a maximum number of cards will be displayed each page
        2. Cards will be ordered by the rating of each location
        3. When a card is clicked, a modal is displayed

Images of the location are displayed

A Description of the location is displayed

Ratings are displayed

The modal will have a button to add to an itinerary they’re building

The modal will have a button to cancel, closing the modal

* + - 1. Once a location is added to an itinerary, a sidebar are the right page will appear
         1. Added locations will be displayed as a descending list of cards

Cards will consist of an image of the location, the name of a location, and a trash icon.

Clicking the trash icon will remove the added location from the sidebar, stopping it from being added to an itinerary

* 1. View Itineraries
     1. Once clicked, the user will be presented a page with all their built itineraries
        1. Itineraries will be displayed as a list of Cards
           1. Cards will consist of the name of the itinerary and the start date and end date of the itinerary
  2. Account
     1. Once clicked, the user will be presented an account settings page

## Feature List

* User Accounts: users can sign up, log in, and save their trip itineraries.
* Trip Management: users can create multiple trips. Can also edit, delete, save, and view their trips.
* Lodging Selection: users can add hotels to a trip and assign the number of nights.
* Dining options: users can add restaurants and pick the specific meal of the day (breakfast, lunch, dinner).
* Attractions and Activities: users can add events, attractions, and activities, and choose the time frame for each.
* Daily Timeline View: trips are displayed in a day-by-day schedule showing the hotels, meals, and activities the user has chosen.
* Mobile Access: The system will be accessible on mobile, so users can use all the features on the go.

o

These are the requirements that you expect to finish by the end of the semester.

It is better to be too detailed rather than not detailed enough. Every team

member should have a clear understanding of what will be built. This section is

an agreement among the team members and the instructor on what must be

accomplished this semester.

o

Create an

\*\*extremely\*\*

detailed list of requirements for every component and

feature of the system.

o

Group related requirements

Use nesting or headings.

o

Every requirement should have a unique numerical-type identifier.

Each item should be a short, specific sentence that explains exactly one

requirement.

o

Do not assume anything about the system; write down all requirements.

o

In general, how the GUI looks is not a requirement unless the system requires

the feature to work.

Not necessary: The OK button will be on the bottom right.

Necessary: Every dialog box displaying an informational message will

have an OK button that closes the dialog when clicked.

6.

Updated Timeline

## Timeline

1. Week 2 (Sep 1-5):
   1. Draft and submit the project proposal (all members)
2. Week 3 (Sep 8-12):
   1. Project requirements document (all members)
   2. Slideshow presentation (all members)
3. Week 4 (Sep 15-19):
   1. Finalize requirements document
   2. Present the slideshow on features
4. Week 5 (Sep 22-26):
   1. Set up project structure(all members)
   2. Set up server on Vercel and database on Supabase(Dustin)
   3. Educating team members on how Vercel and Supabase work and aiding in database setup by pitching ideas for database information(Taylor)
   4. Begin basic account creation and login system (front-end: design and style login UI) (Sorensen)
   5. Implementing Google OAuth
   6. Creating basic navbar (Home, create, view, account)
   7. Build simple API calls in a test to fetch hotels or restaurants by location(Dustin)
5. Week 6 (Sep 29-Oct 3):
   1. Create category sidebar (lodging default) with filter option
   2. View itineraries and create the itineraries page (cards with info for locations)
   3. Implement trip management-CRUD operations (Sorensen)
   4. Develop trip management UI components (create, update, delete trip pages)(Taylor)
   5. Implement user storage for the database(Dustin)
6. Week 7 (Oct 6-10):
   1. Right-side “building” sidebar with a removal option
   2. Implement the lodging selection of adding hotels and assigning nights(Taylor)
   3. Implement dining options of selecting restaurants and the meal assignment (Sorensen)
   4. Integrate API’s into lodging and dining features(Dustin)
   5. Build lodging/dining UI screens and integrate with API results (Sorelle)
   6. Save itinerary to the account feature
7. Week 8 (Oct 13-17):
   1. Implement attractions and activities features with time slots(Taylor)
   2. Add trip data handling, such as trip creation, lodging nights, restaurants, and activities (Sorensen)
   3. Integrate API for attractions and activities lookup(Dustin)
   4. Develop attractions/activities UI and calendar input fields (Sorelle)
8. Week 9 (Oct 20-24):
   1. Implement a timeline view, day-by-day schedule
   2. Connect the timeline view to the database records (Sorensen)
   3. Connect API data to the timeline view(Dustin)
   4. Build interactive timeline/day-by-day itinerary view UI (Sorelle)
9. Week 10 (Oct 27-31):
   1. Review the core features and make adjustments if necessary(all members)
   2. Test database with multiple users and trips(Dustin)
   3. Sorelle: Refine UI/UX consistency (colors, layout, navigation)
10. Week 11 (Nov 3-7):
    1. Implement the extra features if we have time(all members)
    2. Optimize API calls to reduce duplicate calls(Dustin)
    3. Assist with UI integration for extra features (Sorelle)
11. Week 12 (Nov 10-14):
    1. Conduct full system tests(all members)
    2. Write a technical report(all members)
    3. Support usability testing and document UI design choices (Sorelle)
12. Week 13 (Nov 17-21):
    1. Finalize implementation/make sure all features are implemented and working(all members)
    2. Polish the frontend and user experience (Sorelle)
13. Week 14 (Nov 24-28):
    1. Final round of testing(all members)
14. Week 15 (Dec 1-5):
    1. Final presentation(all members)

## BrainStorming Ideas

* Creating user account
* Account Management
  + Deleting user accounts
  + Making updates to account
    - Ex: Changing password
* Sign-In Forms
* Sign-Up Forms
* Sign-Out Option
* Searching for Travel Destination
* Location Hub (Choosing categories and looking up locations around Travel Destination)
  + Lodging
  + Dining
  + Attractions
* Receiving Data on Locations from API’s
* Itinerary Builder
  + Expandable list of locations on a timeline
* Saving Itineraries
* Deleting Itineraries
* Editing Saved Itineraries
* How user information is stored on Database
* How Client communicates with Server

Proposal Recommendations

* Remove Java, use [Node.js](http://node.js)
  + [Node.js](http://node.js) is in JavaScript
    - Rather than getting the project working with Java and JavaScript, you can just use one language
    - Also allows you to use frameworks to get your stuff up and running
* Use React / other frameworks to get Html/CSS working much more quickly
* Spring might be too much for this project
  + Look back at what spring is giving you
* PHP / Node for the server
  + You will find more resources for it
* Other things beside intelliJ
  + VSCode for Web Development
* Why Can’t you use Firebase?
  + If you need logic on the server-side (Code making decisions)
  + The Free side does not require credit card, but does not give you server-side code
  + Spring, PHP, Node will not work on Firebase
* Vercel
  + Not doing Spring with this
  + React / Vue
  + Can do some things for free
* Supabase
  + Database of some sort
  + Don’t know if you can run server-side code
* Do not make your own login service
  + Google has authentication API’s
    - Will do 90% of the work
  + Just pick one (Google, Facebook, etc.)
* Vercel for Server, Supabase for database