Group 32

Jace, Bryce, Kit, Ian, Jenna, K

AI Schedule Builder

This project is a schedule and vacation building application, utilizing AI in order to automate many tasks such as day to day planning, adding data such as weather, travel time and budget, along with suggesting activities to do on your vacation.

Architecture Description

This AI Schedule builder will primarily be based on a modular design, allowing for there to be many independent parts that will all work together seamlessly to create our finished product. This modular design will be be used primarily because of the many benefits that it provides, such as scalability, being able to improve on each sprint release, maintainability, being able to maintain each component separately, and testability, being able to test each component independently which will in turn give the most streamlined development process for our AI Schedule Builder.

The first of these components will be the user interface component, or UI. Within this component, will be everything that the user will actually end up interacting with. This component will begin as a CLI interface, with plans to later convert this into a graphical user interface in later sprints after the core functionality is complete.

From the user interface, users will be able to select from a variety of options. The most important function in the beginning is the creation of events/tasks. The user enters in a name and time for the task and is then presented with some optional fields. They can enter a location, and choose whether the task is fixed or flexible. Fixed tasks are immovable and must be scheduled exactly for the previously chosen duration. Flexible tasks can be moved around to better slot into the daily sequence of events. Furthermore, events can be put into categories for organization.

A SQL database stores all the data relevant to the operation of the scheduler. Tasks are created and stored along with accompanying data such as start time, budget, duration, description, location and more. Data is stored per user, allowing multiple people to use the schedule builder separately. The scheduler will send and retrieve information through SQL queries.

Once tasks are created, the user can then select available tasks and edit their details. In addition, the user can delete tasks, removing them from the schedule and the database. The scheduler also allows for the insertion of dedicated breaks with adjustable durations. Finally, the user can define periods of time as vacations. These vacations can have a duration and a location. New or previously created tasks can be added into the period and become part of the vacation.

The scheduler will feature integration with several APIs that will enhance the vacation functionality. For example, the Google Places API will take the location chosen by the user and suggest points of interest nearby. Forecasted weather data from the location will be collected and displayed if available. If there is a price associated with a point of interest, that will also be

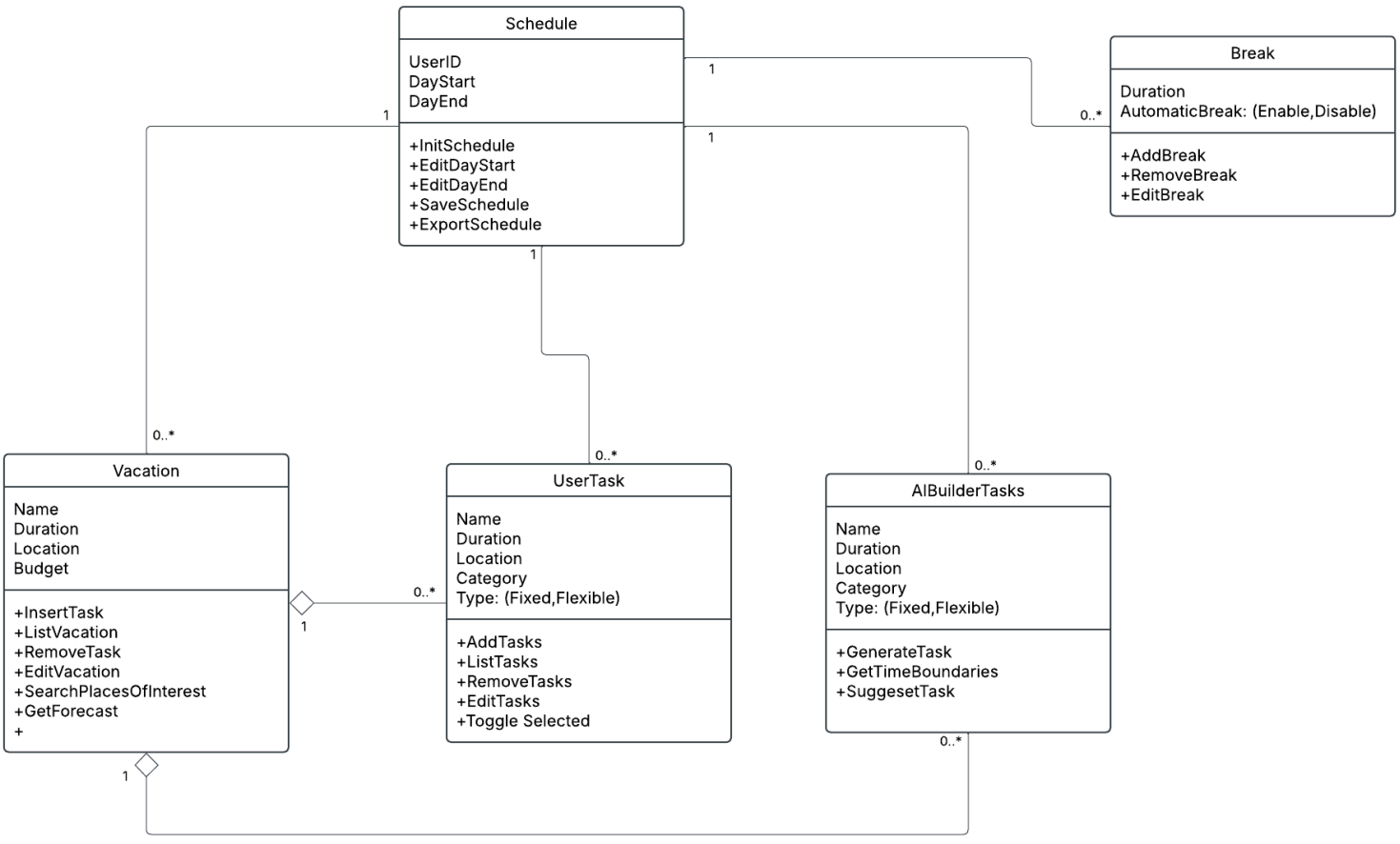
added. Finally, the travel time between events with associated locations will be calculated. The user will be able to enter a budget for the entire trip or for a day and then compare this with the expected cost of the tasks/events scheduled in the tool.

A key component of the scheduler will be AI integration. Once a schedule is built, the AI will be able to see what tasks already exist and what time slots are still open. The user can then select an option to have the AI build some or all of the schedule with generated tasks. These tasks will be suggested with consideration of the time they are being scheduled for. For example, if there is an opening at 8:00 am, the AI would suggest ‘breakfast’ instead of ‘dinner’.

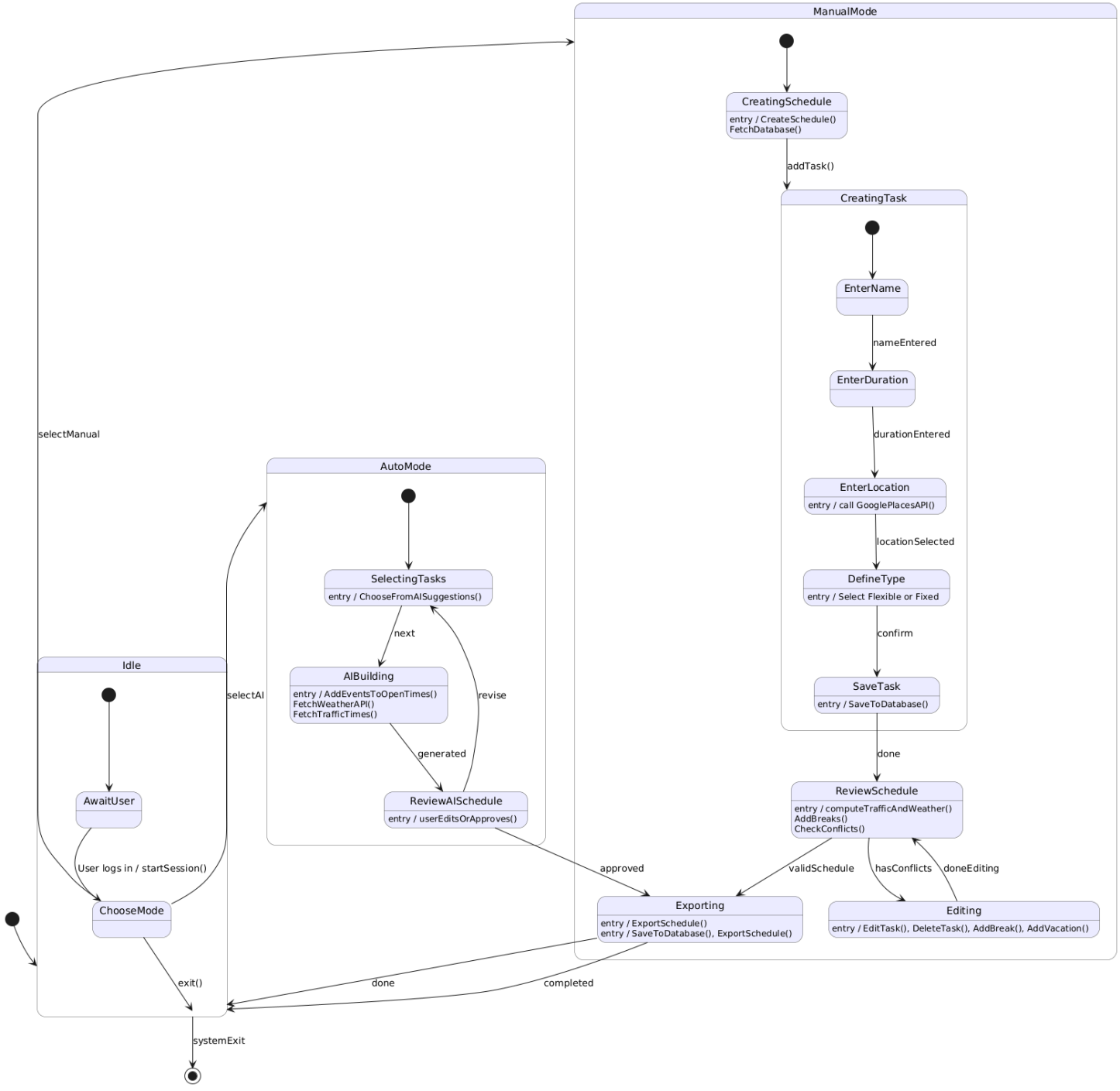
The manual scheduling functionality will have its own module, while the generative functionality will exist in another. The generation module will read from the SQL database to inform its suggestions.

UML Diagrams

UML Class Diagram



UML State Diagram



UML Use Case Diagram