UML Diagrams

The UML diagrams presented in this chapter are simplified representations of the system's architecture. Certain components like classes or use cases have been generalized for clarity and readability. For example, event-related controllers have been unified into a single controller as they share similar logic and structure.

The Use Case Diagram in Figure 3 shows the interaction between the user and the StudyFlow application. The user can perform the action such as adding, updating, and deleting events, viewing different weeks in the calendar, switching between weeks and changing the application language. These use cases reflect the core features of the application.

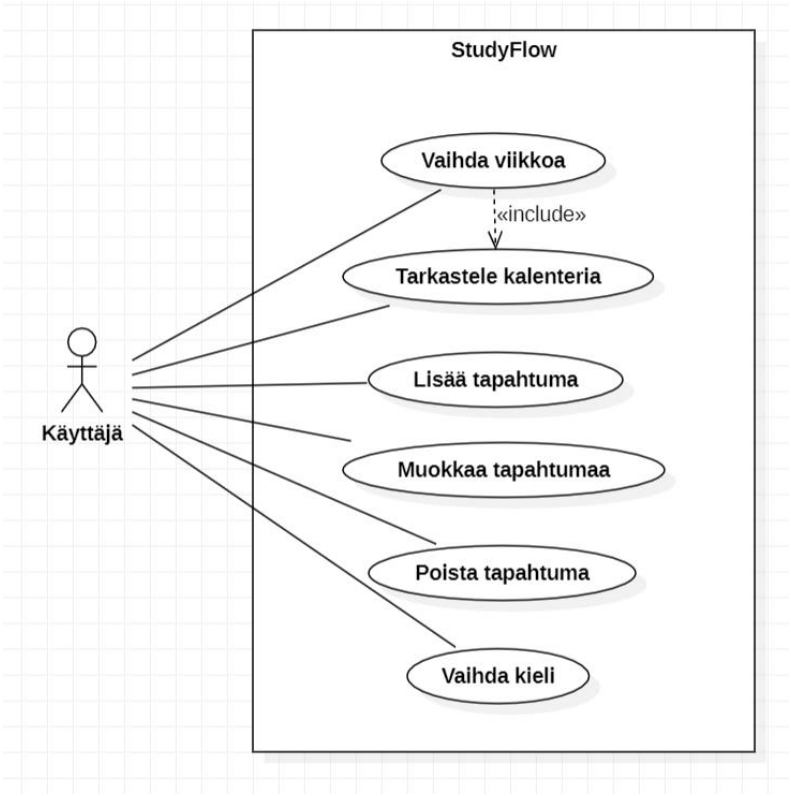

Figure 3: Use Case Diagram of the project.

The Activity Diagram in Figure 4 describes the workflow of adding a new event to the calendar. The process begins when the user clicks the "Add" button. This prompts the application to open a new window for selecting the event type. After the user selects the event type the application opens the selected type's event window. The user is then prompted to fill in the information. After filling in the necessary information, the system validates the input, and this leads to a decision point. If the input was invalid the application asks the user to fix the errors, and the activity goes back to the user filling out the information. If all the inputs were valid the system saves the event to the database and then updates the calendar view.

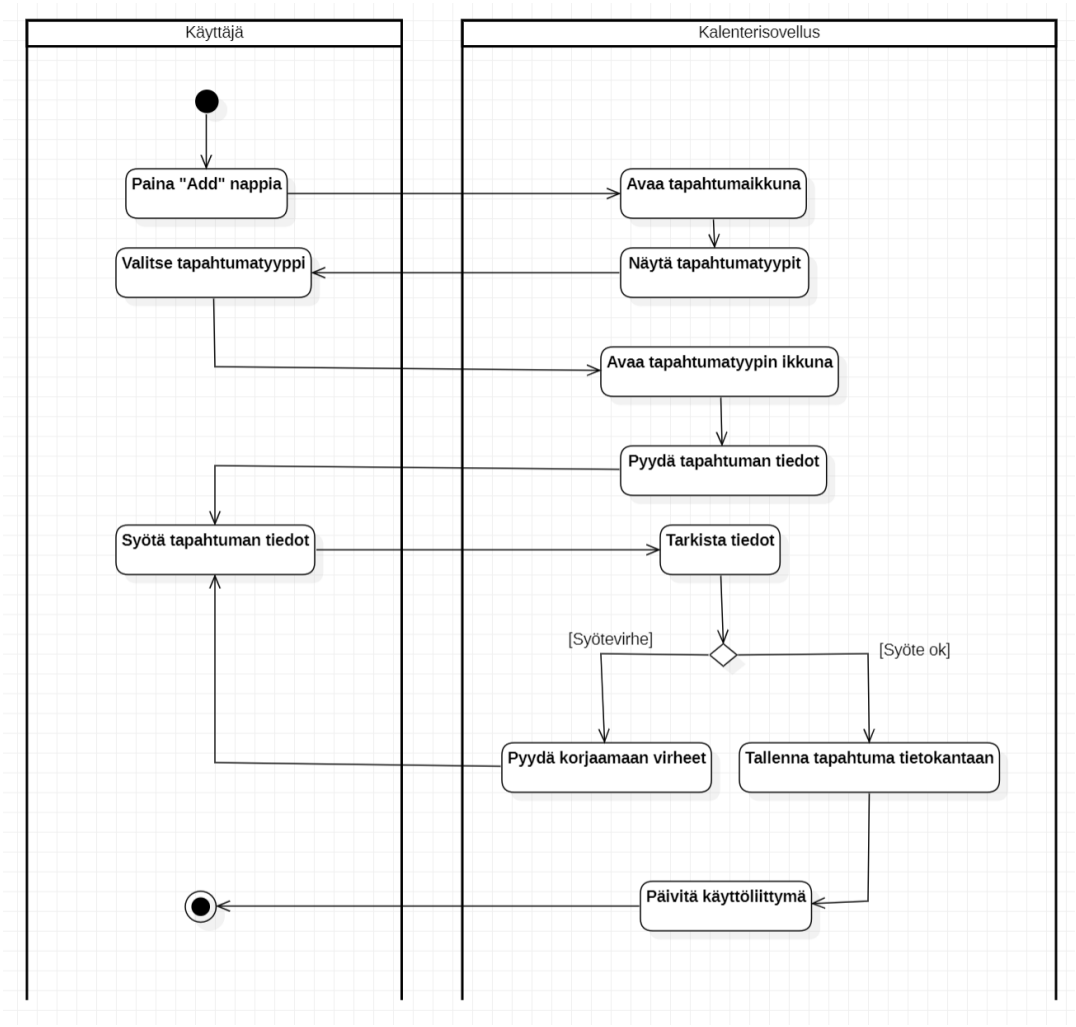

Figure 4: Activity Diagram of the project.

The Sequence Diagram in Figure 5 shows the interaction between components when the user adds an event to the calendar.

1. The user filling out the required event details and clicking the “Add” button.
2. EventController checks that all required fields have been filled, and the input is valid.
3. If any fields are missing or are invalid, an error message is displayed to the user.
4. If the input is valid, the controller calls EventDao to add the event.
5. EventDao then creates a new Event object.
6. EventDao issues a query for adding the event to Database.
7. Upon successful insertion, the database returns confirmation.
8. DAO forwards this confirmation to the EventController.
9. EventController updates the calendar view, and it is presented to the user.

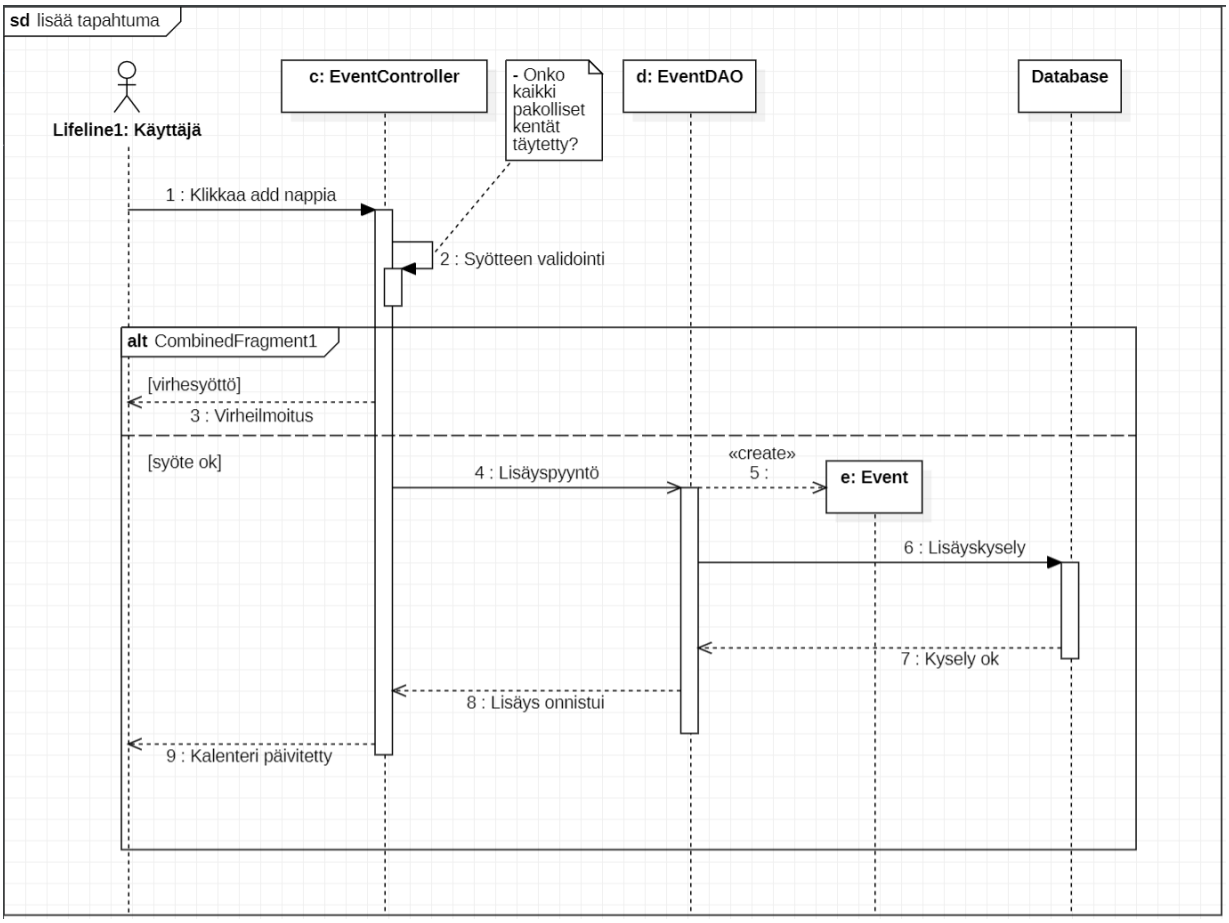

Figure 5: Sequence Diagram of the project.

The Package Diagram in Figure 6 displays the application's package structure. The Main package includes the entry point of the system and imports the contents of the View package. The user interface layer in the View package gets access to the controllers through FXML integration. The Controller package has access to the dao package and imports the Model package. The Dao package imports the Config package and the Model package.

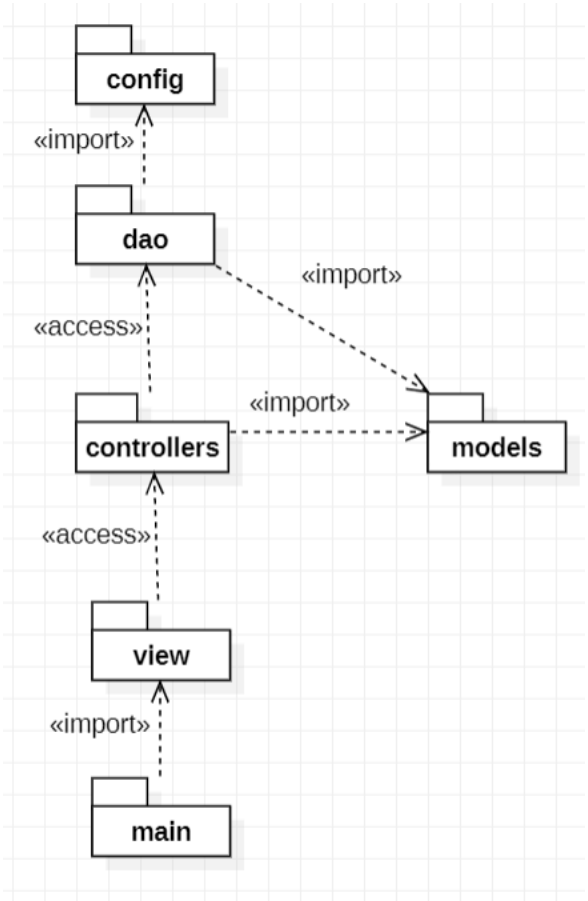

Figure 6: Package Diagram of the project.

Figure 7 presents a simple Class Diagram of the project. It includes the relevant model classes, controller classes, and dao classes, as well as the connections between them. The abstract Event class acts as a base for the event classes. The event classes Assignment, ClassSchedule, Exam, and StudySession inherit the abstract Event class. These classes include specific event's attributes and a reference to a Course class object. The Course class includes course specific information for example a name and an instructor. The TimetableController manages the calendar view. Includes methods for fetching the timetable, creating the event boxes and displaying, updating and deleting an event by calling CourseDAO and EventDAO. It also includes the logic for managing the language setting and translating the application by calling SettingDAO. The AddEventController is responsible for managing adding a course by calling EventDAO. The AddCourseController is responsible for managing adding an event by calling CourseDAO. CourseDAO includes the necessary crud operations for the course and EventDAO includes similar crud operations for the events. With these operations the classes are able to fetch, add, update and delete events and courses into the database. SettingDAO manages fetching, adding and updating the language setting. All the DAO classes call MariaDbConnection class to connect to the database.

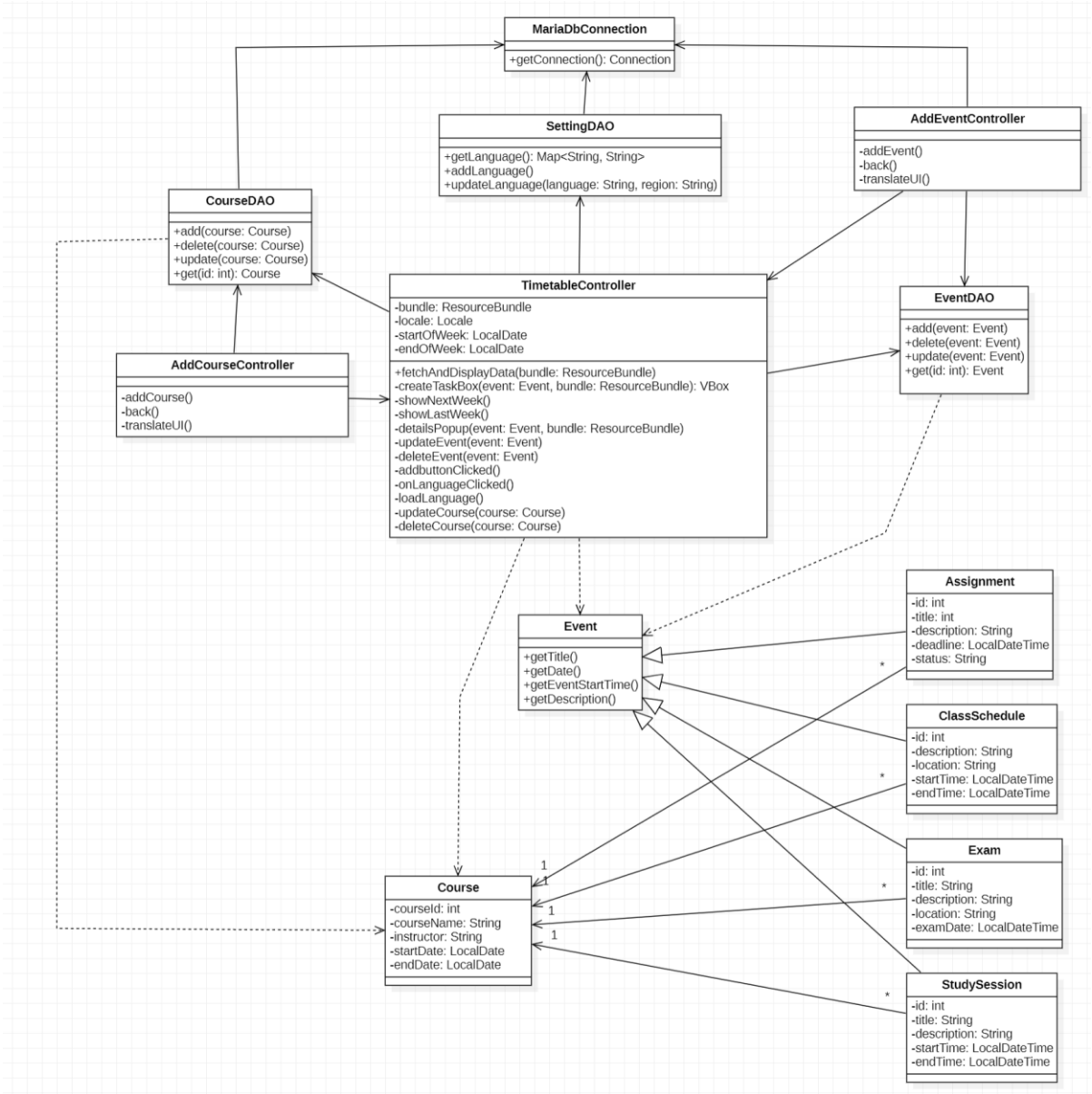

Figure 7: Class Diagram of the project.