Documentation

Team Note-bridge1

Table of content:

Dao (Data Access Object) Classes:

- BookingDAO - CRUD operations for the Booking model class.
    - getBookings() - Returns the list of bookings.
    - insertBookingAlongWithPayment() - Inserts a booking along with the payment, the price is added from the lessonId.
    - getBookingById() - Gets a booking by its id.
    - extractBooking() - Extracts a booking from a result set.
    - deleteBookingById() - Deletes a booking by its id.
    - getBookingsByStudentId() - Returns a list of bookings by a studentId.
    - getBookingList() - Returns a list of bookings used by other methods.
    - getBookingByLessonId() - Returns a list of bookings by a lessonId.
    - getBookingByScheduleId() - Returns a list of bookings by its scheduleId.
    - getBookingCount() - Returns the amount of bookings took by a student.
    - getNumberOfBookings() - Returns the number of bookings.
    - setBookingCancelled() - Changes the booking’s status to cancelled.
    - setBookingFinished() - Changes the booking’s status to finished.
- InstrumentDAO - CRUD operations for the Instrument model class.
    - getInstruments() - Returns the list of bookings from the database.
    - getInstrumentById() - Returns a instrument by its id.
    - getInstrumentCount() - Returns the number of instruments.
    - insertInstrument() - Inserts a new instrument in the database.
    - checkInstrumentByName() - Checks if a instrument with the specified name already exists.
    - deleteInstrument() - Deletes a instrument by its id.
    - updateInstrument() - Updates a instrument with new details.
    - checkInstrumentById() - Checks if a instrument with the specified id already exists.
    - getInstrumentsLearnedByStudent() - Returns a list of instrument learned by a student.
- LessonDAO - CRUD operations for the Lesson model class.
    - getLessonById() - Returns a lesson by its id.
    - extractLesson() - Extracts a lesson from a result set.
    - checkLessonExistsById() - Checks if a lesson exists by an id.
    - getLessons() - Returns all lessons from the database.
    - insertLesson() - Inserts a lesson in the database.
    - deleteLesson() - Deletes a lesson by an id.
    - updateLesson() - Updates a lesson with new details.
    - getLessonBySearch() - Gets a list of lessons with the given search parameters.
    - getLessonArrayList() - Returns an array list of lessons from a result set.
    - getNumberOfLessons() - Returns the number of lessons.
    - getLessonsByStudentId() - Returns a list of lessons by a studentId.
    - getLessonsByTeacherId() - Returns a list of lessons by a teacherId.

- MessageDAO - CRUD operations for the Message model class.
    - getChatHistoryOfUser() - Returns a list of messages that were sent or received by the user.
    - addNewMessage() - Adds a new message in the database and returns the id.
    - deleteMessageById() - Deletes a message by the specified id.
    - getNumberOfMessages() - Returns the number of messages from the database.
- NotificationDAO - CRUD operations for the Notification model class.
    - getNotificationById() - Returns a notification by the given id.
    - extractNotification() - Extracts a notification from a given result set.
    - getNotificationsForUser() - Returns a list of notifications for a given userId.
    - addNotification() - adds a notification to the database and returns its id.
    - confirmNotification() - Updates the notification status to be confirmed.
    - deleteNotification() - Delete a notification by a given id.
    - getNotificationCount() - Returns the number of notifications.
- PaymentDAO - CRUD operations for the Payment model class.
    - updatePaymentToBePayed() - Updates a payment status to be payed.
- ReviewDAO - CRUD operations for the Review model class.
    - getReviews() - Returns a list of the reviews from the database.
    - getReviewById() - Returns a review by the given id.
    - getReviewsOfTeacher() - Returns the reviews of a teacher by the specified teacherId.
    - getReviewsOfStudent() - Returns the reviews of a student by the specified studentId.
    - addReviewToTeacher() - Adds a review to a teacher.
    - getCountOfTeacher() - Gets the number of reviews of a teacher.
    - getCountOfStudent() - Gets the number of reviews of a student.
    - deleteReview() - Deletes a review by a given id.
- SkillDAO - CRUD operations for the Skill model class.
    - getSkills() - Returns all skills from the database.
    - getSkillById() - Returns a skill by the given id.
- TeacherDAO - CRUD operations for the Teacher model class.
    - getTeachers() - Returns the list of teachers from the database.
    - extractTeacher() - Extracts a teacher from the given result set.
    - getTeacherById() - Returns a teacher by the given id.
    - getTeacherCount() - Returns the number of teachers in the database.
    - insertTeacher() - Inserts a new teacher in the database.
    - checkTeacherExistsById() - Checks if a teacher exists by the specified id.
    - deleteTeacher() - Deletes a teacher by the given id.
    - updateTeacherInstruments() - Updates the instruments of a teacher.
    - addTeacherInstrument() - Inserts a new instrument given a teacherId.
    - removeTeacherInstrument() - Removes a instrument given a teacherId.
    - processZipcode() - Returns true if it’s a valid zipcode
- TeacherInstrumentsDAO - CRUD operations for the TeacherInstrument model class
    - getTeacherInstruments() - Returns a list of all the teacher instruments.

    - getInstrumentByTeacherId() - Returns all teacher instruments with the given teacherId.
    - getTeacherByInstrumentId() - Returns all teachers with the given instrumentId.
- TeacherScheduleDAO - CRUD operations for Teacher Schedule model class.
    - getTeacherSchedule() - Returns all teacher schedules.
    - extractSchedule() - Extracts a teacher schedule from a result set.
    - insertTeacherSchedule() - Inserts a new teacher schedule in the database.
    - deleteTeacherSchedule() - Deletes a teacher schedule by the given id.
    - getTeacherSchedulesByTeacherId() - Returns a list of teacher schedules by the given teacherId.
    - getTeacherScheduleById() - Returns a teacher schedule by the given id.
    - getFreeTeacherSchedulesByTeacherId() - Returns a list of teacher schedules that are free by the given teacherId.
    - getTeacherScheduleCount() - Returns the number of teacher schedules.
- UserDAO - CRUD operations for User model class.
    - ReadPepper() - for password security measure.
    - getUsers() - Returns all the users from the database.
    - extractUser() - Extracts a user from the given result set.
    - getUserById() - Returns a user by the given id.
    - insertUser() - insert a new user in the database.
    - hashPasswordArgon2() - Hashes a password using the Argon2 algorithm.
    - verifyPasswordArgon2() - Verifies a password by hashing it with the provided salt and pepper, and comparing to the stored hash.
    - checkUserExistsByEmail() - Checks if a user exists with the specified email.
    - deleteUser() - Deletes the user from database by the given id.
    - updateUser() - Updates a user from database.
    - updateUserDescription() - Updates a new description by the given id.
    - getIdByEmail() - Returns the id associated with the given email.
    - getUserByEmail() - Returns the user associated with the given email.
    - countUsers() - Returns the number of users in the database.
    - countCities() - Returns the distinct amount of cities in the database.
    - setOnline() - Sets the user online by the given id.
    - setOffline() - Sets the user offline by the given id.
- ZipcodeCoordinateDAO - CRUD operations for the Zipcode Coordinate model class.
    - getZipcodeCoordinates() - Retrieves all the zipcodes from the database.
    - getZipcodeByName() - Returns a Zipcode by the given name.
    - isZipcodeExists() - Returns true if the given zip code exists.
    - insertZipcodeCoordinate() - Inserts a new zipcode in the database.
    - updateZipcodeCoordinate() - Updates the zipcode coordinates with a new latitude and longitude.

Resources:

All classes contain methods to communicate with the database, this is done by creating an http request to the database with the prepared statements from the respective DAO classes.
Below are all classes and any comments if methods differ from the explanation above.

- BookingsResource
    - createBookingAndPaymentAndNotification() - adds a booking with the corresponding payment. And a notification for the student and the teacher for the booking confirmation and a notification for the teacher to confirm if the booking has taken place.
    - setBookingCancelled() - checks the CSRF token and sets the booking as cancelled.
    - setBookingFinished() - checks the CSRF token and sets the booking as finished.
- InstrumentsResource
- LessonsResource
    - createLesson() - creates a lesson and checks the CSRF token.
    - deleteLesson() - checks the CSRF token and deletes the lesson.
- MessagesResource
- NotificationResource
    - createNotification() - creates a notification and checks CSRF token.
    - confirmNotification() - checks the CSRF token and confirms the notification.
- PaymentsResource
    - getCountReviewsForLessonFromStudent() - gets the amount of reviews a student has given for a specific lesson.
- ReviewsResource
    - createReview() - creates a review and checks the CSRF token.
    - deleteReview() - checks the CSRF token and deletes the review.
    -
- SkillsResource
- TeacherInstrumentsResource
- TeacherScheduleResource
    - createTeacherSchedule() - creates a schedule (=timeslot) for a teacher and checks the CSRF token.
    - deleteTeacherSchedule() - checks the CSRF token and deletes the schedule.
- TeacherResource
    - searchTeachers() - search teachers by instrument, skill, rating, online or offline and availability date.
    - updateTeacherDetails() - checks the CSRF token and updates the full name, online type, country, city, experience, zip code and instrument of the teacher.
    - checkSessionUser() - checks if the user is currently logged in.

- UsersResource
    - updateUserDetails() - checks the CSRF token and updates the full name, online type, country and city.
    - updateUserDescription - checks the CSRF token and updates the description.
- ZipcodeCoordinatesResource
    - updateZipcodeCoordinate() - updates the current zip code coordinate with new details

Servlets:

- BecomeTeacherServlet:
    - A servlet that allows the creation of a new teacher, if the user is logged in.
- ContactServlet:
    - A servlet used to send an email to the website owners.
- DeleteProfileServlet:
    - A servlet that deletes the profile if the user is logged in.
- LessonsServlet:
    - A servlet to retrieve the lessons.
- LoginServlet:
    - A servlet to handle all login related activities.
- LogoutServlet:
    - A servlet which will log out the user if he is logged in.
- MainServlet:
    - The main servlet.
- MapServlet:
    - A servlet to handle the map.
- MediaServlet:
    - A servlet that can decode media.
- PaymentServlet:
    - A servlet that handles the payment.
- ProfileServlet:
    - A servlet to handle all profile related activities.
- RegisterServlet:
    - A servlet to handle all tasks related to registering.
- UploadServlet:
    - A servlet which handles uploading the (profile) image.

Database

The class Database contains methods to set up the connection with the database.

NominatimAPI

This class contains the method search which gets the coordinates (longitude and latitude) of a location.

Security:

The security class contains methods to sanitise and validate user input.

Javascript:

- Api
  - Constants.js:
    - displayStars - returns the image source string for the amount of stars to show for the average rating.
  - Queries.js: performs CRUD operations through the API.
- Modules
  - calendar.js: makes the calendar interactive.
    - generateCalander - is used to create a calendar on the page.
    - updateCalendar - updates the calendar displayed by the given month and year.
  - filter.js: allows for an interactive filter menu.
    - handleReset - Resets a search input field, checkbox and selected days.
    - loadInstruments - Loads a list of instruments.
  - modal.js: makes the modal interactive.
  - Modal-generator.js: generates the body of the modal
    - fetchTeacherAvailability - fetches and displays the availability of the teacher.
  - nav.js
    - updateNav - nav js dynamically changes the nav bar to show personal notification and profile photo/ chat if authenticated and login/register if not.
  - Notification.js
    - updateNotifications - updates and handles notifications in a container.
    - displayTimePeriod - compares the given date with the current date and checks the time period relative to the current date.
  - Validation.js: (redundant) client-side input validation of all user input. This is also done server-side, but it is added here for redundancy.
- Contact.js: creates the form to allow a user to contact the site administrator.
- Lessons.js: loads and shows available lessons on the lesson page.
  - loadLessons - loads all lessons given certain search parameters.
- main.js: loads the statistics on the main page, and makes the FAQ buttons clickable.
- map.js: contains all logic for displaying teachers on the map and clustering them at lower zoom levels.
  - teacherLoadingAndSearch - loads all teachers given certain search parameters.
- Payment.js: requests the creation of the payment page.
- profile.js: contains all methods to make the profile page interactive.
- register.js: creates the forms for the register page.
- support-chat.js: The full implementation of the support chat functionality is here.

Test classes:

- Dao
  - BookingDAOTest:
    Testing of inserting, deleting and getting in BookingDAO.
  - InstrumentDAOTest:
    Testing of inserting, deleting, checking and updating in InstrumentDAOTest.
  - LessonDAOTest:
    Testing of checking, inserting and deleting in LessonDAO.
  - MessageDAOTest:
    Testing of getting, adding and deleting in MessageDAO.
  - NotificationDAOTest:
    Testing of getting and deleting in NotificationDAO.
  - ReviewDAOTest:
    Testing of adding, deleting and getting in ReviewDAO.
  - SkillDAOTest:
    Testing of getting in SkillDAO.
  - TeacherDAOTest:
    Testing of deleting, getting, inserting and updating in TeacherDAO.
  - TeacherInstrumentDAOTest:
    Testing of getting in TeacherInstruments.
  - TeacherScheduleDAOTest:
    Testing of getting, deleting and inserting in TeacherSchedule.
  - UserDAOTest:
    Testing of getting, deleting, updating and inserting in UserDAO.
  - ZipcodeCoordinateDAOTest:
    Testing of getting, inserting and updating ZipcodeCoordinateDAO.

- Resources
  - GeneralResourceTest:
    Testing of all functionalities of the resource package.

- SecurityTest
  - testRemoveTags - tests a text by removing html tags from an input.