Skill-Based Endorsement Relevance Assessment System(ERAS)

Overview:

The `EndorsementController` class acts as the entry point for handling endorsement-related HTTP requests. It provides endpoints for posting new endorsements and retrieving endorsements for a specific user.

Endpoints:

1. **POST /endorsements:**
   - **Description:** Allows users to post a new endorsement.
   - **Request Body:** Expects a valid `PostEndorsementRequest` object containing the necessary data for the endorsement.
   - **Response:** Returns a `PostEndorsementResponse` object containing information about the posted endorsement.
   - **Validation:** Validates the request body using Bean Validation annotations.
2. **GET /endorsements/{userId}:**
   - **Description:** Retrieves endorsements for a specific user.
   - **Path Variable:** Requires the user ID (`userId`) of the user for whom endorsements are requested.
   - **Response:** Returns a map where keys are skill names and values are lists of formatted endorsement texts.
   - **Validation:** Ensures that the `userId` path variable is not blank.

Dependencies:

- `EndorsementService`: Injected dependency for handling endorsement-related business logic.

Annotations:

- `@RestController`: Indicates that this class is a REST controller.
- `@RequestMapping("/endorsements")`: Specifies the base URL path for all endpoints defined within this controller.
- `@Validated`: Ensures that request data is validated according to Bean Validation annotations.
- `@Autowired`: Injects the `EndorsementService` dependency into the controller.

Logging:

- `logger`: Utilizes SLF4J LoggerFactory for logging. Logs various informational messages, including requests, responses, and other relevant information.

The `EndorsementServiceImpl` class is responsible for handling endorsement-related operations within the application. It provides functionality for posting endorsements, calculating adjusted scores, and retrieving endorsements for a particular user.

Methods:

1. **postEndorsement(PostEndorsementRequest postEndorsementRequest):**
   - This method handles the process of posting an endorsement.
   - It validates the input data, fetches the reviewee, reviewer, and skill entities, calculates the adjusted score, and saves the endorsement to the database.
   - If any required entities are not found or if a duplicate entry is detected, appropriate exceptions are thrown.
2. **hasSkillOrParentSkill(UserEntity user, String skillName):**
   - This method checks whether a user possesses a specific skill or any parent skill.
   - It iterates through the user's skills and recursively checks for a match.
3. **skillMatches(SkillEntity skillEntity, String skillName):**
   - This method checks if a skill entity matches a given skill name.
   - It compares the skill name with the name of the skill entity and its parent skills recursively.

4. **calculateAdjustedScore(EndorsementEntity endorsement, UserEntity reviewer, UserEntity reviewee, SkillEntity skill):**
   - This method calculates the adjusted score for an endorsement based on various criteria.
   - It considers factors such as coworker status, experience level, and skill closeness.
   - The adjusted score and adjustment reasons are then set in the endorsement entity.
5. **areOrWereCoworkers(UserEntity reviewee, UserEntity reviewer):**
   - This method checks if two users are or were coworkers based on their company history.
6. **isReviewerMoreExperienced(UserEntity reviewee, UserEntity reviewer):**
   - This method checks if the reviewer has more experience than the reviewee.
7. **calculatePointsBasedOnSkillCloseness(UserEntity reviewee, SkillEntity skill):**
   - This method calculates additional points based on the closeness of a skill to the reviewee's skills.
8. **getParentSkillCount(SkillEntity skillEntity):**
   - This method calculates the count of parent skills for a given skill.
9. **getEndorsements(String userId):**
   - This method retrieves endorsements for a specific user.
   - It queries the database for endorsements associated with the given user ID and formats the response.
10. **getFormattedEndorsementText(EndorsementEntity endorsement):**

- This method formats the endorsement information into a readable text format.

Dependencies:

- `UserRepository`: Used for accessing user data.
- `SkillRepository`: Used for accessing skill data.
- `EndorsementRepository`: Used for accessing endorsement data.

Exceptions:

- `RevieweeNotFoundException`: Thrown when the reviewee user is not found.
- `ReviewerNotFoundException`: Thrown when the reviewer user is not found.
- `SkillNotFoundException`: Thrown when the skill is not found.
- `ValidationException`: Thrown for validation errors, such as duplicate entries or missing data.

Notes:

- Ensure that all necessary dependencies (`UserRepository`, `SkillRepository`, `EndorsementRepository`) are injected correctly before using this service.
- Handle exceptions appropriately in calling code to provide meaningful error messages to the user.
- This service assumes the availability of the necessary entities (users, skills, endorsements) in the database.