UserInterface
...
...
CourseManagerFacade *
- user: User
- users: UserList
- courses: CourseList
- course: Course
- lesson: Lesson
- exercise: Exercise
+ login(username: String, password: String): User
+ addUserCourse(course: Course): void
+ signUp(username: String, email: String, password: String): void
+ getCourse(uuid: UUID): Course
+ getAllCourses(): CourseList
+ getUsers(): UserList
+ toggleDarkMode(isDarkMode: boolean): void
+ toggleEmailNotification(emailNotification: boolean): void
+ setLesson(index: int): void + getLesson(): Lesson
+ setExercise(index: int): void + getExercise(): Exercise
+ isCorrect(userAnswer: String): boolean + logOut(): void
+ getUserSettings(): Settings
+ displayPosition(): void
+ displayStudyStuff(): void
+ makeStudyFile(): void
+ testStudyStuff(): void
+ playGame(): void
Narrator *
- Narrator(): void
+ playSound(String text): void
- talkPolly(PollyClient polly, String text): void
+ synthesize(PollyClient polly, String text, Voice voice, OutputFormat format): InputStream
VoiceList *
+ showVoices(Region region): void
- displayVoices(PollyClient polly): void
+ main(String[] args): void
Settings *
- emailNotifications: boolean
- isDarkMode: boolean
+ Settings(boolean emailNotif, boolean darkMode)
+ toggleDarkMode(boolean isDarkMode): void
+ getDarkMode(): boolean
+ toggleEmailNotification(boolean emailNotification): void
+ getEmailNotifications(): boolean
<<enumeration>> GoalType *
LESSONS
PROFICIENCY_POINTS
Goal *
- goalType: GoalType
- tracker: int
- goalAmt: int
- isWeekly: bool
+ Goal(GoalType goalType, int goalAmt)
+ getGoalType: GoalType;
+ metGoal(int tracker): boolean
+ updateProgress(boolean addProgress): void
+ currentGoalProgress: int
+ setWeekly(boolean isWeekly: void
+ toString: String
UserList *
- users: ArrayList<User>
- userList: UserList
- UserList()
+ getInstance(): UserList
+ addUser(String username, String email, String password): void
+ editUser(String firstName, String lastName, String email, String password): void
+ getUser(String username, String password): User
+ getUsers(): ArrayList<User>
+ removeUser(String username, String password): void
User *
- username: String
- password: String
- uuid: UUID
- firstName: String
- lastName: String
- emailAddress: String
- courses: HashMap<UUID courseID, int progress>
- proficiencyPointTotal: int
- settings: Settings
- correctWords: WordList
- incorrectWords: WordList
+ User(String firstName, String lastName, String email,...)
+ User(String username, String password, String emailAddress)
+ getIncorrect(): WordList
+ getUsername(): String
+ setWordList(WordList words): void
+ getFirstname(): String
+ getLastName(): String
+ getPassword(): String
+ getEmailAddress(): String
+ getCourse(UUID language): Course
+ getCourseProgress(Course course): int
+ getCourses(): HashMap<Course, Integer>
+ getUUID(): UUID
+ getProfPoints(): int
+ setName(String firstName, String lastName): void
+ setEmailAddress(String emailAddress): void
+ setCourseProgress(ArrayList<int> lessonProgress): void
+ setPassword(String password): void
+ addCourse(Course course): void
+ addCourseProgress(Course course, int lesson, int lessonProgress): void
+ toString(): String
+ isMatch(String username, String password): bool
+ setSettings(Bool darkmodeOn, Bool emailNotifsOn)
+ getSettings(): Settings
- createUUID(): UUID
WordList *
- words: ArrayList<Word>
- wordList: WordList
+ WordList()
+ addWord(String word, String meaning): Word
+ addWord(Word word): void
+ removeWord(Word word): void
+ editWord(String oldWord, String newWord, String newDefinition): void
+ getWord(String searchWord): Word
+ getWords(): ArrayList<Word>
+ setWords(ArrayList<Word> words): void
Word *
- word: String
- meaning: String
- similarWords: ArrayList<Word>
- uuid: UUID
+ Word(String word, String meaning):void
+ getWord():String
+ getMeaning():String
+ speak(); void
+ toString():String
+ setUUID(): UUID
+ getSimilarWords():ArrayList<Word>
+ equals(Object obj): boolean
+ hashCode(): int
CourseList *
- courses: ArrayList<Course>
- courseList: CourseList
- CourseList()
+ getInstance(): CourseList
+ addCourse(Language language, ArrayList<Lesson> lessons)
+ getCourse(UUID uuid): Course
+ getCourse(Language language): Course
+ getCourse(int index): Course
+ toString(): String
Course *
- language: Language
- lessons: ArrayList<Lesson>
- proficiencyPts: int
- uuid: UUID
+ Course (Language language, ArrayList<Lesson> lessons)
+ Course (Language language, UUID uuid, ArrayList<Lesson> lessons)
+ getLesson(int Index): Lesson
+ getLessons(): ArrayList<Lesson>
+ getLanguage(): Language
+ getUuid(): Uuid
+ toString(): String
Lesson *
- subject: String
- intro: String
- exercises: ArrayList<Exercise>
- keyWords: Word[]
- keyPhrases: Phrase[]
- progress: int
+ Lesson(String subject, ArrayList<Excerise> exercises, Word[] words, Phrase[] phrases, int progress)
+ getExercise(Int index): Exercise
+ getExercises(): ArrayList<Exercise>
+ generateExercise(): Exercise
+ getSubject(): String
+ getIntro(): String
+ getProgress(): int
+ toString(): String
Phrase *
- phrase: String
- translation: String
- responses: ArrayList<String>
- phraseWithBlank: String
- missingWord: String
+ Phrase(String phrase, String translation): void
+ Phrase(String phrase, String translation, ArrayList<String> responses, String phraseWithBlank, String missingWord): void
+ getPhrase(): String
+ getTranslation(): String
+ getResponses(): ArrayList<String>
+ getMissingWord(): String
+ toString(): String
DataConstants
...
DataReadWriter *
+ getUsers(): ArrayList<User>
+ updateUsers(ArrayList<User>): void
+ getCourses(): ArrayList<Course>
UserInfo JSON *
[{ "uuid" "firstname" "lastname" "username" "email" "password" "ProfPoints" "settings": { "emailNotifications" "darkMode" } "courses": [{ "language_uuid" "progress" }] "words":[] }]
CourseInfo JSON *
"course": { "language" "uuid" "progress" "lessons": [{ "subject" "progress" "intro" "keywords": [] "keyphrases":[] "questions": [{ "type" "question" "options" "answer" }] }] }
<<enumeration>> Language *
SPANISH
FRENCH
JAPANESE
Exercise *
+ isCorrect(String useranswer): bool
+ toString(): String
+ getWord(): Word
Matching *
# questions: String
# options: String
# answer: Word
# words: Word[]
+ Matching(Words[] words)
+ isCorrect(String useranswer): bool
+ toString(): String
+ getWord(): Word
Fill_In *
# phrase: Phrase
# question: String
# translation: String
# answer: String
+ Fill_In(Phrase question)
+ isCorrect(String useranswer): bool
+ toString(): String
+ getWord(): Word
Conversation *
# phrase: Phrase
# question: Phrase
+ Conversation(String question, String answer)
+ isCorrect(String useranswer): bool
+ toString(): String
+ getWord(): Word
Audio *
# options: ArrayList<Word>
# answer: Word
+ Audio(Word[] words)
+ getWord(): Word
+ isCorrect(String useranswer): bool
+ toString(): String
Translation *
# answer: Word
+ Translation(Word word)
+ isCorrect(String useranswer): bool
+ toString(): String
+ getWord(): Word