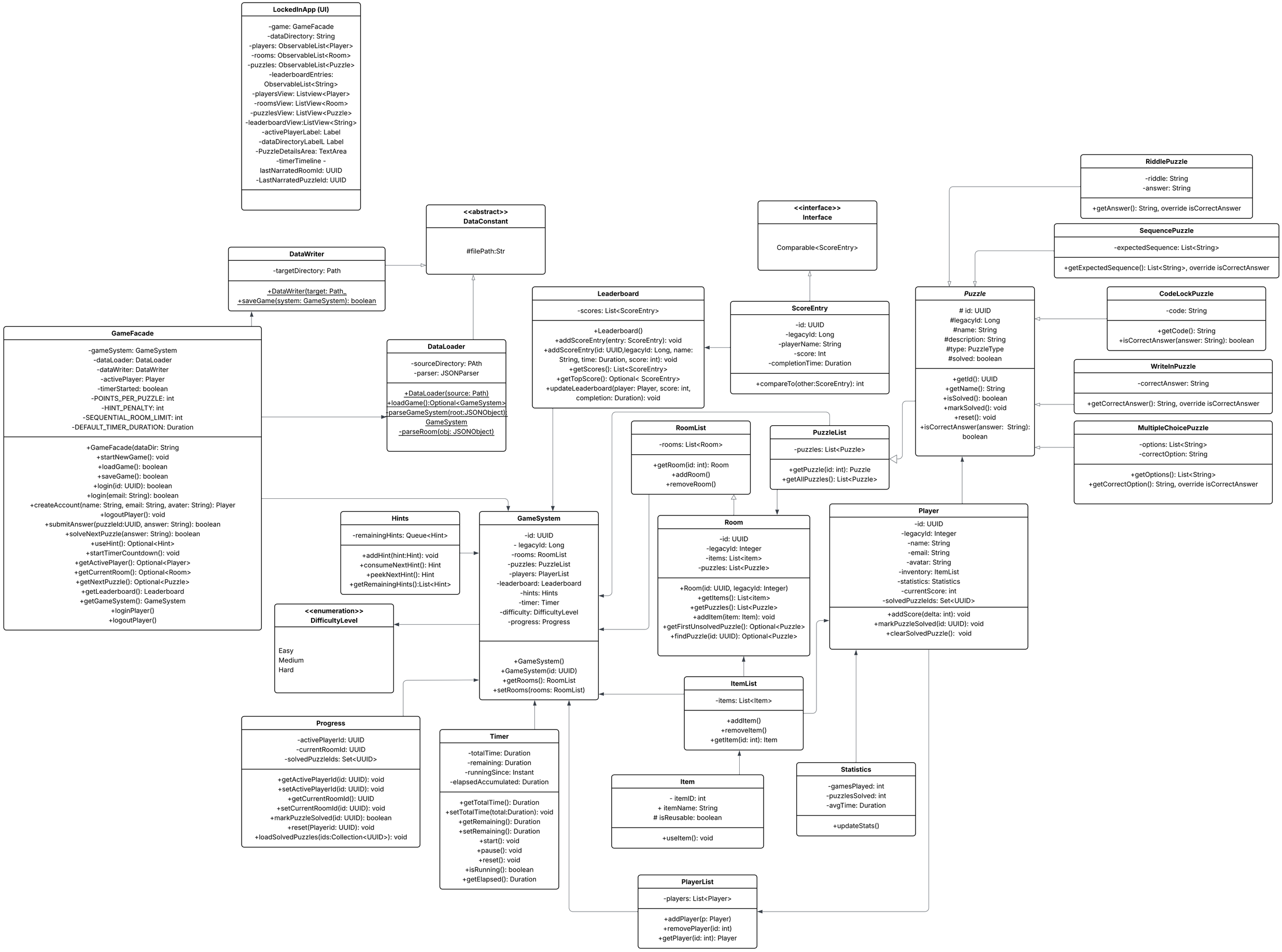
LockedInApp (UI)
-game: GameFacade
-dataDirectory: String
-players: ObservableList<Player>
-rooms: ObservableList<Room>
-puzzles: ObservableList<Puzzle>
-leaderboardEntries:
ObservableList<String>
-playersView: Listview<Player>
-roomsView: ListView<Room>
-puzzlesView: ListView<Puzzle>
-leaderboardView:ListView<String>
-activePlayerLabel: Label
-dataDirectoryLabelL Label
-PuzzleDetailsArea: TextArea
-timerTimeline -
lastNarratedRoomId: UUID
-LastNarratedPuzzleId: UUID
<<abstract>>
DataConstant
#filePath:Str
DataWriter
-targetDirectory: Path
+DataWriter(target: Path_
+saveGame(system: GameSystem): boolean
<<interface>>
Interface
Comparable<ScoreEntry>
RiddlePuzzle
-riddle: String
-answer: String
+getAnswer(): String, override isCorrectAnswer
SequencePuzzle
-expectedSequence: List<String>
+getExpectedSequence(): List<String>, override isCorrectAnswer
Leaderboard
-scores: List<ScoreEntry>
+Leaderboard()
+addScoreEntry(entry: ScoreEntry): void
+addScoreEntry(id: UUID,legacyId: Long, name:
String, time: Duration, score: int): void
+getScores(): List<ScoreEntry>
+getTopScore(): Optional< ScoreEntry>
+updateLeaderboard(player: Player, score: int,
completion: Duration): void
ScoreEntry
-id: UUID
-legacyId: Long
-playerName: String
-score: Int
-completionTime: Duration
+compareTo(other:ScoreEntry): int
Puzzle
id: UUID
#legacyId: Long
#name: String
#description: String
#type: PuzzleType
#solved: boolean
+getId(): UUID
+getName(): String
+isSolved(): boolean
+markSolved(): void
+reset(): void
+isCorrectAnswer(answer: String):
boolean
CodeLockPuzzle
-code: String
+getCode(): String
+isCorrectAnswer(answer: String): boolean
WriteInPuzzle
-correctAnswer: String
+getCorrectAnswer(): String, override isCorrectAnswer
MultipleChoicePuzzle
-options: List<String>
-correctOption: String
+getOptions(): List<String>
+getCorrectOption(): String, override isCorrectAnswer
GameFacade
-gameSystem: GameSystem
-dataLoader: DataLoader
-dataWriter: DataWriter
-activePlayer: Player
-timerStarted: boolean
-POINTS_PER_PUZZLE: int
-HINT_PENALTY: int
-SEQUENTIAL_ROOM_LIMIT: int
-DEFAULT_TIMER_DURATION: Duration
+GameFacade(dataDir: String
+startNewGame(): void
+loadGame(): boolean
+saveGame(): boolean
+login(id: UUID): boolean
+login(email: String): boolean
+createAccount(name: String, email: String, avater: String): Player
+logoutPlayer(): void
+submitAnswer(puzzleId:UUID, answer: String): boolean
+solveNextPuzzle(answer: String): boolean
+useHint(): Optional<Hint>
+startTimerCountdown(): void
+getActivePlayer(): Optional<Player>
+getCurrentRoom(): Optional<Room>
+getNextPuzzle(): Optional<Puzzle>
+getLeaderboard(): Leaderboard
+getGameSystem(): GameSystem
+loginPlayer()
+logoutPlayer()
DataLoader
-sourceDirectory: PAth
-parser: JSONParser
+DataLoader(source: Path)
+loadGame():Optional<GameSystem>
-parseGameSystem(root:JSONObject):
GameSystem
-parseRoom(obj: JSONObject)
RoomList
-rooms: List<Room>
+getRoom(id: int): Room
+addRoom()
+removeRoom()
PuzzleList
-puzzles: List<Puzzle>
+getPuzzle(id: int): Puzzle
+getAllPuzzles(): List<Puzzle>
Hints
-remainingHints: Queue<Hint>
+addHint(hint:Hint): void
+consumeNextHint(): Hint
+peekNextHint(): Hint
+getRemainingHints():List<Hint>
GameSystem
-id: UUID
- legacyId: Long
-rooms: RoomList
-puzzles: PuzzleList
-players: PlayerList
-leaderboard: Leaderboard
-hints: Hints
-timer: Timer
-difficulty: DifficultyLevel
-progress: Progress
+GameSystem()
+GameSystem(id: UUID)
+getRooms(): RoomList
+setRooms(rooms: RoomList)
Room
-id: UUID
-legacyId: Integer
-items: List<item>
-puzzles: List<Puzzle>
+Room(id: UUID, legacyId: Integer)
+getItems(): List<item>
+getPuzzles(): List<Puzzle>
+addItem(item: Item): void
+getFirstUnsolvedPuzzle(): Optional<Puzzle>
+findPuzzle(id: UUID): Optional<Puzzle>
Player
-id: UUID
-legacyId: Integer
-name: String
-email: String
-avatar: String
-inventory: ItemList
-statistics: Statistics
-currentScore: int
-solvedPuzzleIds: Set<UUID>
+addScore(delta: int): void
+markPuzzleSolved(id: UUID): void
+clearSolvedPuzzle(): void
<<enumeration>>
DifficultyLevel
Easy
Medium
Hard
ItemList
-items: List<Item>
+addItem()
+removeItem()
+getItem(id: int): Item
Progress
-activePlayerId: UUID
-currentRoomId: UUID
-solvedPuzzleIds: Set<UUID>
+getActivePlayerId(id: UUID): void
+setActivePlayerId(id: UUID): void
+getCurrentRoomId(): UUID
+setCurrentRoomId(id: UUID): void
+markPuzzleSolved(id: UUID): boolean
+reset(Playerid: UUID): void
+loadSolvedPuzzles(ids:Collection<UUID>): void
Timer
-totalTime: Duration
-remaining: Duration
-runningSince: Instant
-elapsedAccumulated: Duration
+getTotalTime(): Duration
+setTotalTime(total:Duration): void
+getRemaining(): Duration
+setRemaining(): Duration
+start(): void
+pause(): void
+reset(): void
+isRunning(): boolean
+getElapsed(): Duration
Item
- itemID: int
+ itemName: String
# isReusable: boolean
+useItem(): void
Statistics
-gamesPlayed: int
-puzzlesSolved: int
-avgTime: Duration
+updateStats()
PlayerList
-players: List<Player>
+addPlayer(p: Player)
+removePlayer(id: int)
+getPlayer(id: int): Player