BloodLevelsDataGenerator
- baselineCholesterol: double[]
- baselineWhiteCells: double[]
- baselineRedCells: double[]

+ BloodLevelsDataGenerator(patientCount: int)
+ generate(patientId: int, outputStrategy: OutputStrategy): void

BloodPressureDataGenerator
- lastSystolicValues: int[]
- lastDiastolicValues: int[]

+BloodPressureDataGenerator(patientCount: int)
+ generate(patientId: int, outputStrategy: OutputStrategy): void

ECGDataGenerator
- lastEcgValues: double[]
- PI: double

+ ECGDataGenerator(patientCount: int)
+ generate(patientId: int, outputStrategy: OutputStrategy): void
- simulateEcgWaveform(patientId: int, lastEcgValue: double): double

BloodSaturationDataGenerator
- lastSaturationValues: int[]

+ BloodSaturationDataGenerator(patientCount: int)
+ generate(patientId: int, outputStrategy: OutputStrategy): void

AlertGenerator
- RANDOM_GENERATOR: Random
- alertStates: boolean[]

+ AlertGenerator(patientCount: int)
+ generate(patientId: int, outputStrategy: OutputStrategy): void

Generate blood preasure data
Generate blood preassure data for each patient
Generate ECG data for each patient
Gnearte Blood Saturation data for each patient
Generate alerts for each patient

1..*

<<Interface>>
PatientDataGenerator

+ generate(patientId: int, outputStrategy: OutputStrategy): void

HealthDataSimulator
- patientCount: int
- scheduler: ScheduledExecutorService
- outputStrategy: OutputStrategy
- random: Random

+ main(args: String[]): void
+ parseArguments(args: String[]): void
+ printHelp(): void
- initializePatientIds(patientCount: int):   List<Integer>
- scheduleTasksForPatients(patientIds: List<Integer>): void
- scheduleTask(task: Runnable, period: long, timeUnit: TimeUnit): void

uses

<<Interface>>
OutputStrategy

output(patientID: int, timeStamp: long, label: string, data: string): void

uses as default output startegy

WebSocketOutputStrategy
- server: WebSocketServer

+ WebSocketOutputStrategy(port: int)
+ output(patientId: int, timestamp: long, label: String, data: String): void

FileOutputStrategy
- baseDirectory: String
- fileMap: ConcurrentHashMap<String, string>

+ FileOutputStrategy(baseDirectory: String)
+ output(patientId: int, timeStamp: long, label: String, data: String): void

ConsoleOutputStrategy

+ output(patientId: int, timestamp: long,label: String, data: String): void

TcpOutputStrategy
- serverSocket: ServerSocket
- clientSocket: Socket
- out: PrintWriter

+ TcpOutputStrategy(port: int)
+ output(patientId: int, timestamp: long, label: String, data: String): void

SimpleWebSocketServer

+ SimpleWebSocketServer(address: InetSocketAddress)
+ onOpen(conn: WebSocket, handshake: ClientHandshake): void
+ onClose(conn: WebSocket, code: int, reason: String, remote: boolean): void
+ onMessage(conn: WebSocket, message: String): void
+ onError(conn: WebSocket, ex: Exception): void
+ onStart(): void

realize
is