src/../optimise

OptimiseSchedule

+ OPTIMIZE_REASON: String
+ TIME_LIMIT: long
+ OFFSET: long
+ FIFTEEN_MINUTES: long
+ DUMMY_ID: int
+ MINUTE: long
+ MAX_TIME: Timestamp

+ getManualVessels(Map<Integer, List<Schedule>>): List<Vessel>
+ getDummySchedule(int, Timestamp, Timestamp): Schedule
+ removeAutomatics(Map<Integer, List<Schedule>>): Map<Integer, List<Schedule>>
+ sortBerthsOnMinTime(Map<Berth, Timestamp>): List<Berth>
+ getScheduleWithoutDummies(Map<Integer, List<Schedule>>): Map<Integer, List<Schedule>>
+ optimisePlanning(Timestamp, Timestamp, int, boolean, boolean, String): void
+ scheduleTimeInMillis(Berth, Vessel): long
+ getFirstClosedTime(Berth, List<Schedule>, Timestamp): Timestamp
+ updateDatabase(Map<Integer, List<Schedule>>, boolean, String): void
+ getNewScheduleWithManuals(Map<Integer, List<Schedule>>, List<Berth>): Map<Integer, List<Schedul
+ planAutomaticVessel(Vessel, Schedule): Schedule
+ getBestVessel(List<Vessel>, Berth, Timestamp): Vessel
+ getAutomaticVessels(Map<Integer, List<Schedule>>): List<Vessel>
+ possiblyDelay(Timestamp, Time, Time): Timestamp
+ fittingVessels(Berth, List<Vessel>, Timestamp, Timestamp): List<Vessel>
+ getEmptyNewSchedule(List<Berth>): Map<Integer, List<Schedule>>
+ getBestSchedule(Vessel, Berth, Timestamp, Timestamp): Schedule
+ impossibleToSchedule(List<Vessel>, Timestamp): boolean
+ getFirstOpenTime(Berth, List<Schedule>, Timestamp): Timestamp
+ getMinimumTimes(Map<Integer, List<Schedule>>, List<Berth>, Timestamp): Map<Berth, Timestamp>
+ getCurrentTime(): Timestamp

src/../models

Vessel

+ deadline: Timestamp
+ length: int
+ depth: int
+ id: int
+ arrival: Timestamp
+ containers: int
+ name: String
+ destination: int
+ width: int
+ costPerHour: double

+ setDeadline(Timestamp): void
+ setArrival(Timestamp): void
+ getContainers(): int
+ setDepth(int): void
+ toString(Vessel): String
+ setContainers(int): void
+ getDepth(): int
+ equals(Object): boolean
+ getId(): int
+ getName(): String
+ getWidth(): int
+ getCostPerHour(): double
+ setName(String): void
+ getArrival(): Timestamp
+ setCostPerHour(double): void
+ setLength(int): void
+ hashCode(): int
+ getDeadline(): Timestamp
+ getDestination(): int
+ getLength(): int
+ setId(int): void
+ setDestination(int): void
+ setWidth(int): void
+ compareTo(Vessel): int

Change

+ date: Timestamp
+ vesselName: String
+ author: String
+ undo: boolean
+ oldObject: String
+ reason: String
+ newObject: String
+ type: String
+ vessel: Integer
+ id: long

+ getNewObject(): String
+ isUndo(): boolean
+ setUndo(boolean): void
+ equals(Object): boolean
+ getId(): long
+ getAuthor(): String
+ getVessel(): Integer
+ getType(): String
+ getDate(): Timestamp
+ setDate(Timestamp): void
+ setReason(String): void
+ setVessel(Integer): void
+ getReason(): String
+ setId(long): void
+ hashCode(): int
+ getVesselName(): String
+ setNewObject(String): void
+ getOldObject(): String
+ setOldObject(String): void
+ setAuthor(String): void
+ setType(String): void
+ setVesselName(String): void

Berth

+ width: int
+ unloadSpeed: double
+ depth: int
+ open: Time
+ terminalId: int
+ id: int
+ length: int
+ close: Time

+ getUnloadSpeed(): double
+ compareTo(Berth): int
+ isClosedDuring(Schedule): boolean
+ setOpen(Time): void
+ getClose(): Time
+ hashCode(): int
+ equals(Object): boolean
+ setId(int): void
+ getTerminalId(): int
+ setClose(Time): void
+ setTerminalId(int): void
+ getOpen(): Time
+ getWidth(): int
+ setDepth(int): void
+ fits(Vessel): boolean
+ getLength(): int
+ getId(): int
+ setUnloadSpeed(double): void
+ getDepth(): int
+ setWidth(int): void

ScheduleChange

+ date: Timestamp
+ reason: String
+ newSchedule: Schedule
+ author: String
+ vessel: Integer
+ id: long
+ undo: boolean
+ oldSchedule: Schedule

+ getOldSchedule(): Schedule
+ setUndo(boolean): void
+ getVessel(): Integer
+ setDate(Timestamp): void
+ equals(Object): boolean
+ setReason(String): void
+ getDate(): Timestamp
+ setAuthor(String): void
+ setVessel(Integer): void
+ setNewSchedule(Schedule): void
+ getId(): long
+ setId(long): void
+ getAuthor(): String
+ isUndo(): boolean
+ setOldSchedule(Schedule): void
+ getReason(): String
+ getNewSchedule(): Schedule
+ hashCode(): int

VesselChange

+ date: Timestamp
+ author: String
+ reason: String
+ oldVessel: Vessel
+ undo: boolean
+ id: long
+ vessel: Integer
+ newVessel: Vessel

+ getNewVessel(): Vessel
+ getId(): long
+ setAuthor(String): void
+ setOldVessel(Vessel): void
+ setUndo(boolean): void
+ getReason(): String
+ isUndo(): boolean
+ equals(Object): boolean
+ getDate(): Timestamp
+ getVessel(): Integer
+ hashCode(): int
+ getOldVessel(): Vessel
+ setId(long): void
+ setDate(Timestamp): void
+ setVessel(Integer): void
+ setReason(String): void
+ setNewVessel(Vessel): void
+ getAuthor(): String

Schedule

+ start: Timestamp
+ expectedEnd: Timestamp
+ vessel: int
+ manual: boolean
+ berth: int

+ toString(): String
+ getExpectedEnd(): Timestamp
+ setExpectedEnd(Timestamp): void
+ toString(Schedule): String
+ setStart(Timestamp): void
+ compareTo(Schedule): int
+ hashCode(): int
+ getVessel(): int
+ getStart(): Timestamp
+ setVessel(int): void
+ equals(Object): boolean
+ setBerth(int): void
+ getBerth(): int
+ isManual(): boolean
+ setManual(boolean): void

EmployeeInput

+ email: String
+ terminal: Integer
+ password: String
+ port: Integer
+ role: Role

+ getTerminal(): Integer
+ setEmail(String): void
+ hashCode(): int
+ setRole(Role): void
+ generate(): Employee
+ getEmail(): String
+ getRole(): Role
+ equals(Object): boolean
+ getPort(): Integer
+ getPassword(): String
+ setPassword(String): void
+ setTerminal(Integer): void
+ compareTo(EmployeeInput): int
+ setPort(Integer): void

Employee

+ key: SaltedKey
+ email: String
+ role: Role
+ terminal: Integer
+ port: Integer

+ getEmail(): String
+ setKey(SaltedKey): void
+ getRole(): Role
+ getPort(): Integer
+ compareTo(Employee): int
+ getKey(): SaltedKey
+ hashCode(): int
+ setEmail(String): void
+ getTerminal(): Integer
+ equals(Object): boolean
+ setPort(Integer): void
+ setTerminal(Integer): void

Infeasibility

+ valid: boolean
+ type: String
+ overlapping: List<Integer>
+ OTHER: String
+ reason: String
+ OVERLAPPING: String

+ getOverlapping(): List<Integer>
+ getType(): String
+ setType(String): void
+ setValid(boolean): void
+ isValid(): boolean
+ setOverlapping(List<Integer>): void
+ getReason(): String

Performance

+ scheduledVessels: int
+ totalCost: double
+ unscheduledVessels: int

+ getScheduledVessels(): int
+ setScheduledVessels(int): void
+ getTotalCost(): double
+ setTotalCost(double): void
+ setUnscheduledVessels(int): void

Port

+ id: int
+ name: String

+ getId(): int
+ setId(int): void
+ hashCode(): int
+ compareTo(Port): int
+ setName(String): void
+ getName(): String

<<enumeration>> Role

+ VESSEL_PLANNER:
+ value: String
+ PORT_AUTHORITY:
+ clearance: int
+ RESEARCHER:
+ TERMINAL_MANAGER:

+ getValue(): String
+ compare(Role, Role): int
+ values(): Role[]
+ valueOf(String): Role
+ fromValue(String): Role

Terminal

+ portId: int
+ name: String
+ id: int

+ getPortId(): int
+ getId(): int
+ setName(String): void
+ setId(int): void
+ getName(): String
+ compareTo(Terminal): int
+ setPortId(int): void
+ equals(Object): boolean
+ hashCode(): int

src/../type_adapters

TimestampAdapter

+ formatter: DateTimeFormatter

+ unadaptOrElse(String, Timestamp): Timestamp
+ unadapt(String): Timestamp
+ adapt(Timestamp): String

CustomJacksonJsonProvider

+ mapper: ObjectMapper

+ createMapper(): ObjectMapper

TimeAdapter

+ adapt(Time): String
+ unadapt(String): Time

src/../util

TokenUtil

+ SECRET_KEY_ALGORITHM: String
+ SECURE_RANDOM_ALGORITHM: String
+ SECRET: String
+ SECRET_SIZE: int
+ BASE64_DECODER: Decoder
+ SECURE_RANDOM_PROVIDER: String
+ JSON: JsonNodeFactory
+ SECRET_KEY_PROVIDER: String
+ BASE64_ENCODER: Encoder

+ validateToken(String): Response
+ validate(HttpServletRequest): Response
+ generateSignature(Token): String
+ generateSecret(): String
+ generateToken(Employee): String
+ invalidRequest(String): Response
+ invalidToken(String): Response
+ main(String[]): void

Permission

+ vessel(Vessel): Supplier<Permission>
+ employee(Employee): Supplier<Permission>
+ check(Employee): boolean
+ schedule(Schedule): Supplier<Permission>
+ checkInternal(Employee): boolean
+ terminal(Terminal): Supplier<Permission>
+ vessel(int): Supplier<Permission>
+ berth(int): Supplier<Permission>
+ role(Role): Supplier<Permission>
+ berth(Berth): Supplier<Permission>
+ port(int): Supplier<Permission>
+ employee(EmployeeInput): Supplier<Permission>
+ terminal(int): Supplier<Permission>
+ port(Port): Supplier<Permission>
+ schedule(int): Supplier<Permission>
+ employee(String): Supplier<Permission>

EncryptionUtil

+ SECRET_KEY_PROVIDER: String
+ SALT_SIZE: int
+ MD5: MessageDigest
+ SECURE_RANDOM_PROVIDER: String
+ SECURE_RANDOM_ALGORITHM: String
+ SECRET_KEY_ALGORITHM: String
+ PBE_KEY_ITERATION_COUNT: int
+ PBE_KEY_LENGTH: int

+ bytesToHex(byte[]): String
+ outputProviders(): void
+ validate(String, SaltedKey): boolean
+ generateKey(String): SaltedKey
+ main(String[]): void
+ generateSalt(): byte[]

Configuration

+ inTestEnvironment: boolean

+ useTestEnvironment(boolean): void
+ getProperty(String): String

test/../dao

DummyData

+ TERMINALS: List<Terminal>
+ BERTHS: List<Berth>
+ VESSEL_CHANGES: List<VesselChange>
+ PORTS: List<Port>
+ SCHEDULE_CHANGES: List<ScheduleChange>
+ SCHEDULES: List<Schedule>
+ VESSELS: List<Vessel>
+ EMPLOYEES: List<Employee>

+ main(String[]): void
+ createDummyData(): void

DaoTest

+ MIN_TIME: Timestamp
+ MAX_TIME: Timestamp

+ scheduleChangeTest(): void
+ replaceTest(): void
+ vesselChangeTest(): void
+ setup(): void
+ createGetTest(): void
+ deleteTest(): void

GetTest

+ main(String[]): void

test/../util

EncryptionUtilTest

+ ITERATION_COUNT: int
+ RANDOM: Random

+ random(): String
+ fuzzTest(): void

TestUtil

+ listEqualsIgnoreOrder(List<T>, List<T>): boolean

test/../models

ModelTest

+ main(String[]): void

src/../dao

GenericDao

+ MIN_TIME: Timestamp
+ MAX_TIME: Timestamp
+ connection: Connection

+ getConnection(): Connection
+ performOperations(Operations<R>): R

ChangeDao

+ undoLastChange(String): int
+ redoLastChange(String): int
+ getChangesByTerminal(int, Timestamp, Timestamp): List<Change>

VesselChangeDao

+ createVesselChange(VesselChange): VesselChange?
+ getVesselChangesByVessel(int, Timestamp, Timestamp): List<VesselChange>
+ getVesselChanges(Timestamp, Timestamp): Map<Integer, List<VesselChange>>
+ deleteVesselChanges(int, Timestamp, Timestamp): int
+ deleteVesselChangeByAuthor(String, Timestamp): int
+ fromResultSet(ResultSet): VesselChange
+ getVesselChangeByDate(int, Timestamp): VesselChange?
+ deleteVesselChangeByDate(int, Timestamp): int

ScheduleDao

+ isOverlapping(Schedule): Infeasibility?
+ deleteScheduleByVessel(int): int
+ replaceSchedule(int, Schedule): Schedule
+ getScheduleByVessel(int): Schedule?
+ getSchedulesByBerth(int, Timestamp, Timestamp): List<Schedule>
+ getSchedulesByTerminal(int, Timestamp, Timestamp): Map<Integer, List<Schedule>>
+ getSchedulesByPort(int, Timestamp, Timestamp): Map<Integer, Map<Integer, List<Sche

ScheduleChangeDao

+ createScheduleChange(ScheduleChange): ScheduleChange?
+ deleteScheduleChanges(int, Timestamp, Timestamp): int
+ fromResultSet(ResultSet): ScheduleChange
+ deleteScheduleChangeByDate(int, Timestamp): int
+ getScheduleChangesByVessel(int, Timestamp, Timestamp): List<ScheduleChange>
+ getScheduleChanges(Timestamp, Timestamp): Map<Integer, List<ScheduleChange>>
+ deleteScheduleChangeByAuthor(String, Timestamp): int
+ getScheduleChangeByDate(int, Timestamp): ScheduleChange?

TerminalDao

+ getTerminalsByPort(int): Map<Integer, Termina
+ createTerminal(Terminal): Terminal
+ deleteTerminal(int): int
+ replaceTerminal(int, Terminal): int
+ getTerminal(int): Terminal?

BerthDao

+ deleteBerth(int): int
+ replaceBerth(int, Berth): int
+ getBerth(int): Berth?
+ getBerthsByTerminal(int): Map<Integer, Berth>
+ createBerth(Berth): Berth

PortDao

+ deletePort(int): int
+ createPort(Port): Port
+ importPort(JsonNode): Port?
+ replacePort(int, Port): int
+ getPort(int): Port?
+ getPorts(): Map<Integer, Port>
+ exportPort(int, Timestamp, Timestamp): String?

EmployeeDao

+ replaceEmployee(String, Employee): int
+ createEmployee(Employee): Employee
+ fromResultSet(ResultSet): Employee
+ getInteger(ResultSet, String): Integer?
+ getEmployees(): List<Employee>
+ deleteEmployee(String): int
+ getGravatar(String): JsonNode
+ getEmployee(String): Employee?

VesselDao

+ getVessel(int): Vessel?
+ replaceVessel(int, Vessel): int
+ createVessel(Vessel): Vessel
+ deleteVessel(int): int
+ getCountUnscheduledVesselsByTerminal(int, Timestamp, Timestamp): int
+ getVesselsByTerminal(int, Timestamp, Timestamp): Map<Integer, Vessel>
+ getUnscheduledVesselsByTerminal(int, Timestamp, Timestamp): List<Vessel>

src/../resources

TerminalResource

+ uriInfo: UriInfo
+ id: int
+ request: Request

+ deleteTerminal(HttpServletRequest): Response
+ getTerminal(HttpServletRequest): Response
+ getChanges(HttpServletRequest, String, String): Response
+ getVessels(HttpServletRequest, String, String): Response
+ getPerformance(HttpServletRequest, String, String): Response
+ replaceTerminal(HttpServletRequest, Terminal): Response
+ getUnscheduledVessels(HttpServletRequest, String, String): Response
+ getSchedules(HttpServletRequest, String, String): Response
+ getInfeasibilites(HttpServletRequest, String, String): Response
+ getBerths(HttpServletRequest): Response

VesselResource

+ request: Request
+ id: int
+ uriInfo: UriInfo

+ replaceSchedule(HttpServletRequest, Schedule): Response
+ deleteSchedule(HttpServletRequest): Response
+ getInfeasibility(Schedule): Infeasibility?
+ deleteVessel(HttpServletRequest): Response
+ replaceVessel(HttpServletRequest, Vessel): Response
+ isFeasible(HttpServletRequest): Response
+ getSchedule(HttpServletRequest): Response

PortResource

+ id: int
+ request: Request
+ uriInfo: UriInfo

+ exportPort(HttpServletRequest, String, String): Response
+ getPerformance(HttpServletRequest, String, String): Response
+ deletePort(HttpServletRequest): Response
+ getTerminals(HttpServletRequest): Response
+ replacePort(HttpServletRequest, Port): Response
+ getPort(HttpServletRequest): Response

ScheduleChangeResource

+ uriInfo: UriInfo
+ request: Request
+ vessel: int

+ getScheduleChanges(HttpServletRequest, String, String): Response
+ deleteScheduleChanges(HttpServletRequest, String, String): Response
+ getScheduleChangeByDate(HttpServletRequest, String): Response

VesselChangeResource

+ uriInfo: UriInfo
+ request: Request
+ vessel: int

+ deleteVesselChanges(HttpServletRequest, String, String): Response
+ getVesselChanges(HttpServletRequest, String, String): Response
+ deleteVesselChangeByDate(HttpServletRequest, String): Response

EmployeesResource

+ request: Request
+ uriInfo: UriInfo

+ createEmployee(HttpServletRequest, EmployeeInput): Response
+ getEmployees(HttpServletRequest): Response

TokenResource

+ verify(HttpServletRequest): Response
+ obtain(HttpServletRequest, LoginRequest): Response

TerminalsResource

+ uriInfo: UriInfo
+ request: Request

+ createTerminal(HttpServletRequest, Terminal): Response
+ getTerminal(String): TerminalResource

VesselsResource

+ uriInfo: UriInfo
+ request: Request

+ createVessel(HttpServletRequest, Vessel): Response
+ getVessel(String): VesselResource

UndoResource

+ uriInfo: UriInfo
+ request: Request

+ undoLastChange(HttpServletRequest): Response
+ redoLastChange(HttpServletRequest): Response

VesselChangesResource

+ request: Request
+ uriInfo: UriInfo

+ getVesselChange(String): VesselChangeResource
+ getVesselChanges(HttpServletRequest, String, String): Response

PortsResource

+ uriInfo: UriInfo
+ request: Request

+ importPort(HttpServletRequest, String): Response
+ getPort(String): PortResource
+ createPort(HttpServletRequest, Port): Response
+ getPorts(HttpServletRequest): Response

EmployeeResource

+ request: Request
+ email: String
+ uriInfo: UriInfo

+ replaceEmployee(HttpServletRequest, EmployeeInput): Response
+ getGravatar(HttpServletRequest): Response
+ deleteEmployee(HttpServletRequest): Response
+ getEmployee(HttpServletRequest): Response

BerthsResource

+ uriInfo: UriInfo
+ request: Request

+ getBerth(String): BerthResource
+ createBerth(HttpServletRequest, Berth): Response

ScheduleChangesResource

+ uriInfo: UriInfo
+ request: Request

+ getScheduleChange(String): ScheduleChangeResource
+ getScheduleChanges(HttpServletRequest, String, String): Response

BerthResource

+ uriInfo: UriInfo
+ request: Request
+ id: int

+ replaceBerth(HttpServletRequest, Berth): Response
+ getSchedules(HttpServletRequest, String, String): Response
+ getBerth(HttpServletRequest): Response
+ deleteBerth(HttpServletRequest): Response