# Song Tianyi - Project Portfolio

## PROJECT: Deliveria

---

## Overview

Deliveria is a **desktop application** that allows a **delivery manager** to **manage and assign delivery tasks** efficiently. While it consists of a *Graphical User Interface* (GUI) that is user-friendly, Deliveria is **optimized for those who prefer** to work with a *Command Line Interface* (CLI) which allows fast management of the delivery tasks in an organisation.

## Summary of contributions

- **Major enhancement**: added the feature to **schedule delivery tasks**, with **automatic suggestions based on the driver's existing schedule**
  - What it does: allows the user to assign a driver to a task, and schedule this task to be completed in the future, without causing time clashes with the existing tasks of the driver
  - Justification: This is a core feature of the product. By enabling this feature, we help the managers to better utilise the human resources, save time in planning the task, and deliver the orders with satisfaction
  - Highlights: The command will automatically suggest a better time slot, so as to ensure that all drivers' schedules are the most optimised. At the same time, we are giving the managers the freedom to exercise their own judgement, and override the automatic stuggestions
  - Credits: The choice of a backing data structure is inspired by a CS2040 Practical Exam question
- **Minor enhancement**:
  - [TBC] added a feature to correct the user's spelling mistakes in a command, by suggesting the most similar command word
  - changed three existing classes to generic classes, so as to reduce 600+ lines of repetitive code
- **Code contributed**: [[Functional code]] [Test code] *{give links to collated code files}*
- **Other contributions**: [TBC]
  - Project management:
    - Managed releases `v1.3` - `v1.5rc` (3 releases) on GitHub
  - Enhancements to existing features:
    - Updated the GUI color scheme (Pull requests #33, #34)
    - Wrote additional tests for existing features to increase coverage from 88% to 92% (Pull requests #36, #38)

- Documentation:
  - Did cosmetic tweaks to existing contents of the User Guide: #14
- Community:
  - PRs reviewed (with non-trivial review comments): #12, #32, #19, #42
  - Contributed to forum discussions (examples: 1, 2, 3, 4)
  - Reported bugs and suggestions for other teams in the class (examples: 1, 2, 3)
  - Some parts of the history feature I added was adopted by several other class mates (1, 2)
- Tools:
  - Integrated a third party library (Natty) to the project (#42)
  - Integrated a new Github plugin (CircleCI) to the team repo

*{you can add/remove categories in the list above}*

# Contributions to the User Guide

*Given below are sections I contributed to the User Guide. They showcase my ability to write documentation targeting end-users.*

### Assign a task to a driver: `assign`

Assign an existing task a driver with a proposed time.
Format: `assign [force] t/TASK_ID d/DRIVER_ID at/hMM - hMM`

- You cannot assign a time that clashes with the driver's existing schedule, or is outside their working hours
- If there is an earlier time slot that the driver is available, the program will suggest the earlier time slot
  - Use `assign force` to dismiss the suggestion and add the proposed time to the task
- The ID is a positive integer

Examples:

- `assign t/1 d/1 at/900-1200`
  Schedule task #1 from 11 am to 12 pm, and assign it to driver #1.
- `assign force t/2 d/1 at/1600-1700`
  Schedule task #2 from 4 pm to 5 pm, and assign it to driver #1 regardless the existence of an earlier time slot.

### Unschedule a task: `free`

Remove the time slot and driver from a task, and free the driver from this time slot in their

schedule.
Format: `free t/TASK_ID`

> - You cannot free a task that has no driver or time slot assigned to it.

Examples:

- `free t/1`
  Remove the assigned driver and time slot from the task, and free the driver's schedule.

# Contributions to the Developer Guide

*Given below are sections I contributed to the Developer Guide. They showcase my ability to write technical documentation and the technical depth of my contributions to the project.*

## Task Scheduling

### Design Considerations

- A `Schedule` should be a collection of non-overlapping `EventTime` object, and is always sorted
- Should be able to notify the user if a better time slot is available, while giving users the liberty to exercise their own judgments

### Implementation

Every `Driver` keeps track of a `Schedule` class, which is backed by a naturally sorted, TreeSet of `EventTime` objects.

Before a new `EventTime` is added to the schedule, the method checks against the set of object to ensure the addition will not result in overlapping `EventTime` in the schedule. This operation works in logarithmic time thanks to the tree structure.

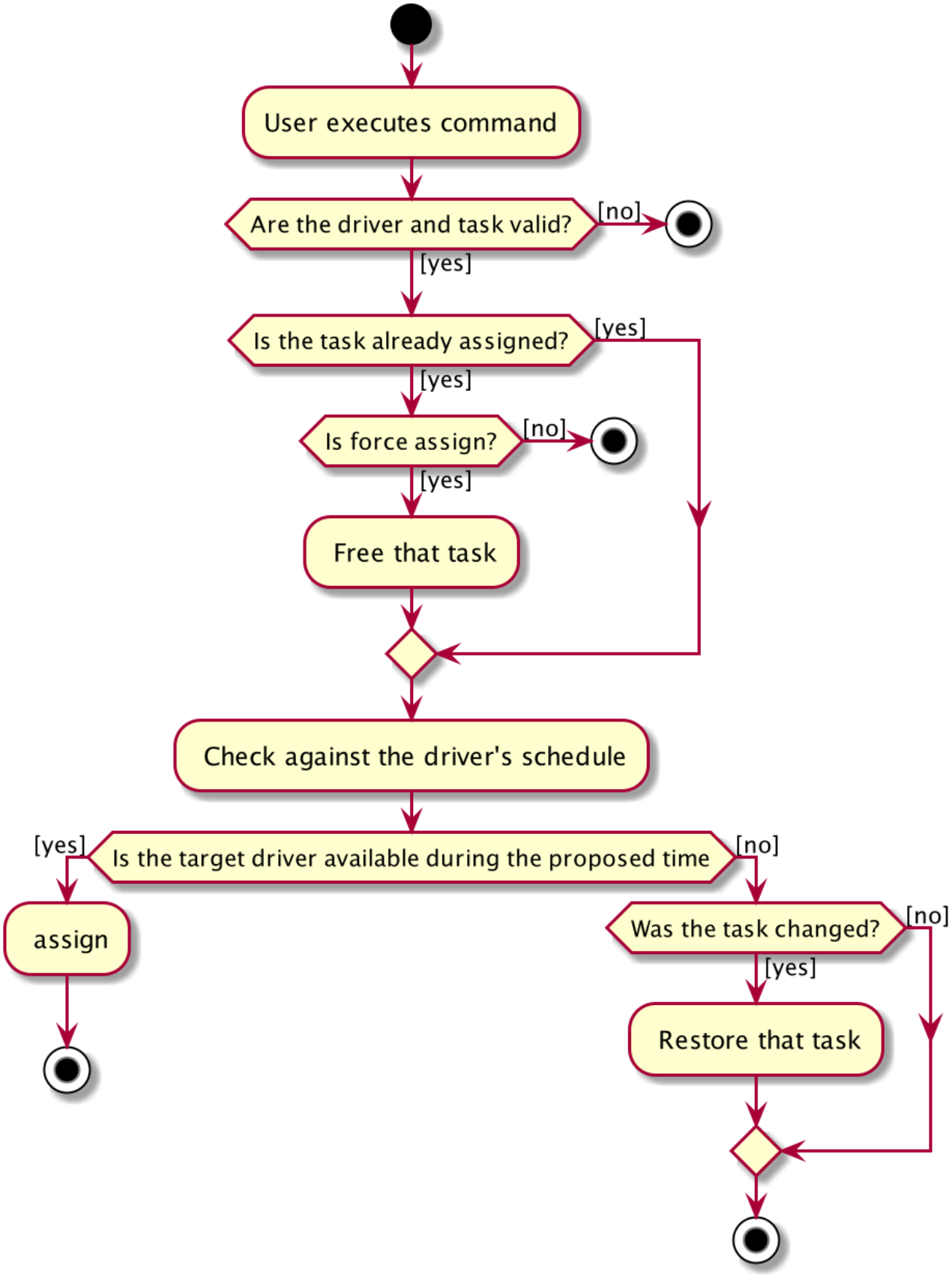
In order to better utilise a driver, we implement a method to suggest an earlier alternative time slot in a schedule. When adding a time to a schedule, this method will:

- Calculate the duration of proposed `EventTime`
- Perform a linear greedy search in the schedule, to find the first slot that can fit the duration

| NOTE | Since the schedule guarantees no overlapping `EventTime`, there is no complication in handling the start and end times. |
|---|---|

In order to enforce the optimised scheduling method above, the program will block every `assign` command that has a suboptimal proposed time, unless the user uses the `force` argument. Moreover, the `assign` and `free` command are the only commands that modify the `Driver` and `EventTime` attributes of a `Task`, so that all drivers will have an optimised schedule, unless `force assign` is used.

The following activity diagram summarizes the checks happened when user executes an `assign` command.




After the above checks has passed, `assign` command will:

- Set the `Driver` and `EventTime` attributes in the `Task`

- Add the proposed `EventTime` to the `Driver's Schedule`

Similarly, calling `free` command will:

- Remove the existing `EventTime` from the `Driver's Schedule`
- Set the `Driver` and `EventTime` attributes to `null`

# PROJECT: PowerPointLabs

*{Optionally, you may include other projects in your portfolio.}*