Part i)

The continuous integration method we have used for the second period of the assessment is GitHub Actions, namely "*Java CI with Gradle*". We felt this was appropriate for our project as it is directly linked to the version control method we used (GitHub) and was therefore easy to implement into the repository. Actions allows us to automate workflows and makes it easier to build a CI pipeline. GitHub Actions also makes it easier to review code and find errors that need to be fixed or things that can be tweaked. The approach we will be using is to use Actions to automate repetitive and tedious tasks that would take our attention away from the main project development.

Part ii)

```yaml
name: Java CI with Gradle

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

permissions:
  contents: read

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v3
    - name: Set up JDK 11
      uses: actions/setup-java@v3
      with:
        java-version: '11'
        distribution: 'temurin'
    - name: Build with Gradle
      uses: gradle/gradle-build-action@v2.3.3
      with:
        arguments: build
```

The continuous integration infrastructure we have used is shown the code snapshot here. What this does is automatically build the project everytime we commit to GitHub, whilst also reporting if said build is successful or not. If it is the latter, the CI will show a red cross and if the former a green tick. Due to it being an automated system, we only have to check each commit by looking at the symbol generated thus saving a lot of time when developing the code for our project,