Git branching strategy

* **What version control strategy did you choose and how did you actually implement/enforce it?**  
  We used Git with a lightweight workflow. Instead of following a formal branching model such as GitFlow or feature branching, we primarily worked directly on the main branch. Since our group practiced pair programming throughout the project, we committed changes once a task or feature was completed. In some cases, this meant smaller incremental commits, while in others the commits were larger and contained multiple changes. We initially attempted to follow specific commit message conventions, but these were not always consistently applied.
* **Why did your group choose this approach over others?**  
  We selected this strategy because it allowed us to work more easily as a group without needing to frequently resolve merge conflicts or coordinate separate branches. Working directly on the master branch aligned well with our approach to pair programming, as we were always in sync and did not need to spend extra time explaining or reviewing changes that had been done independently. We therefore considered this method simpler and more efficient for our way of working.
* **What advantages and disadvantages did you encounter?**  
  The main advantage was that our workflow was straightforward and minimized the risk of merge conflicts, as all development was carried out together. This kept the group aligned at all times and reduced the overhead of branch management. However, a significant disadvantage was that our progress could be slowed down when we were not able to meet as a group. Since our workflow depended on pair programming and committing changes together, development paused whenever one or more members were unavailable.