Development Plan
Software Engineering

Team 8, RLCatan
Matthew Cheung
Sunny Yao
Rebecca Di Filippo
Jake Read

Table 1: Revision History

Date	**Developer(s)**	**Change**
Sept. 22	Sunny,Rebecca	inital dev plan created for milestone 1

This document outlines the development plan for RLCatan, an AI agent that learns to play the board game *Catan* competitively using reinforcement learning.

1 Confidential Information

There is no confidential information to protect in this project.

2 IP to Protect

There is no IP to protect in this project.

3 Copyright License

AlgoCatan is adopting the MIT License, which can be found at this link.

4 Team Meeting Plan

Our team regularly meets 4:30-6:30pm every Thursday. We will schedule additional meetings every time we have an addressable agenda. Virtual meetings will be held through a Discord call and physical meetings will be held at tutorial locations or ETB. Meetings may be hybrid depending on schedules. We will schedule meetings with our advisor when we require resources or expertise. Meetings will be structured with a set agenda devised by the notetaker before scheduling to address all of the topics we need to discuss.

5 Team Communication Plan

We will be using Discord as our main communication platform. We will create a server with channels for general discussion, meeting scheduling, and project management. For more formal communication, we will use email to contact our advisor. We will also use GitHub Issues for tracking existing issues and their closures.

6 Team Member Roles

- Team Leader(Jake): this person is responsible for scheduling meetings, ensuring deadlines are met, and overall team coordination. This person will also be the main point of contact with the supervisor and the TA.
- Notetaker(Rebecca): this person is responsible for creating meeting agendas and also taking notes during meetings. This person will also be updating the Kanban board.

- IT(Sunny): this person is responsible for managing the GitHub repository, including branches, pull requests, and issues. This person will also be resonsible for trouble-shooting any technical issues that arise.
- Researcher(Matthew): this person is responsible for researching any topics that the team is unfamiliar with. This person will also be responsible for finding relevant papers and articles to help the team understand the project better.

As the project progresses, these roles may shift depending on individual strengths and interests. If any roles are too demanding, we will consider redistributing tasks to ensure a balanced workload.

7 Workflow Plan

We will be using GitHub for version control and collaboration. To manage development, we will create branches for each task on the Kanban Board, along with sub-branches for individual team members work. Should this prove infeasible, we may deviate from this fine-grained strategy, opting to group multiple items in single pull requests. Pull requests will be used to review and merge code changes, ensuring quality and consistency. GitHub Issues will be used to track tasks and bugs, with issues assigned to specific team members or whoever is available. Issue categories will be managed via tags, labelling them as front-end, back-end, heuristics, etc. To streamline this process, we will also use issue templates for consistency in reporting and classification of issues. We will be using CI for automated testing. For CD, we will develop deployment pipelines in the event that we have access to a computing node where we can train our RL agent. We also plan to utilize SonarQube for static code analysis.

8 Project Decomposition and Scheduling

We will be using GitHub Projects to manage our project tasks and track milestones. Our project Kanban board will have columns for "To Do", "In Progress", "In Review", and "Done", with each task represented as a card that moves across the columns as it progresses. Tasks and milestones will be scheduled according to the deadlines outlined in the course, ensuring that the team meets all deliverable requirements on time. The teams Notetaker is resonsible for keeping this board up to date. Large tasks will be broken down into smaller, manageable subtasks to make progress easier to track and estimate. Task assignments will be based on each team member's strengths, experience, and availability. To use our resources realistically, we will estimate the time required for each task, monitor progress. If needed we will adjust the schedule or reassign tasks.

The link to our GitHub project board can be found here.

The major milestones for our project are as follows:

Milestone	Due Date
Team Formed, Project Selected	Sept. 15
Problem Statement, POC Plan, Development Plan	Sept. 22
Requirements Document & Hazard Analysis Revision	Oct. 6
V&V Plan Revision	Oct. 27
Design Document Revision	Nov. 10
Proof of Concept Demonstration	Nov. 17
Design Document Revision	Jan. 19
Revision 0 Demonstration	Feb. 2
V&V Report & Extras Revision 0	March 9
Final Demonstration (Revision 1)	March 23
EXPO Demonstration	TBA
Final Documentation (Revision 1)	April 6

Table 2: Project Milestones and Due Dates

9 Proof of Concept Demonstration Plan

The main risks for the success of our project are the AI not being able to effectively learn to play *Catan* at a high level, and the user interface not being intuitive or user-friendly. To mitigate these risks, we will conduct regular Elo testing against existing benchmark opponents to make sure the AI is improving or at least not regressing. We will also conduct user testing sessions to gather feedback and make iterative improvements to the interface. Lastly we will ensure that our AI models are thoroughly tested and validated before deployment. When we demonstrate our proof of concept, we will showcase that the AI is able to read gamestate and return valid moves in *Catan*, even if it is not performing at a high level at this stage.

10 Expected Technology

We expect to use Python as our primary programming language, as well as using the Catanatron open source libary for simulating *Catan* games and training our models. We will also utilize React and JavaScript as our web framework for building the user interface and digital twin. For computer vision models, we will be using openCV and yolov9 for image recognition and processing. We will be using GitHub Actions to implement CI for automated testing and potentially CD for deployment pipelines.

11 Coding Standard

Since our projects backend is primarily in Python, we will be following the PEP 8 Coding Standard. The link to this standard can be found here.

Our frontend is primarily in JavaScript and React, so we will be following the google style guide coding standard. The link to this standard can be found here.

Appendix — Reflection

1. Why is it important to create a development plan prior to starting the project?

2. In your opinion, what are the advantages and disadvantages of using CI/CD?

3. What disagreements did your group have in this deliverable, if any, and how did you resolve them?

Jake Read:

1. A development plan helps formulate our expectations for the work in the coming months. It allows us to pre-plan our strategies for various milestones and encourages us to look ahead into what technology we'll be using at each design stage. I feel it goes hand in hand with the problem statement in regard to preventing scope creep, as it details an initial outline we planned to follow from initiation of the project. Early decisions regarding elements such as the coding standard we'll use will ensure that our design philosophy remains consistent throughout the project and prevent conflicting styles. Additionally, establishing confidential info, IPs to protect, copyright licenses, etc. as early as possible is critical for making decisions going forward.

2. The main advantage of CI/CD in my opinion is automated testing. Having the full test suite run on new builds can help catch bugs much earlier and leads to improved code quality. It's also faster to release new builds once all the pipelines are set up, since the process is fully automated, and this can also improve deployment consistency since there's less human error. As for the disadvantages, while some of us have used CI/CD in co-ops, none of us have actually built any CI/CD pipelines before, so it will take significant effort to set it all up, and we're likely to make mistakes that will cause deployment issues. It's still probably worth it though, since using it is the only way we'll get that experience.

3. We did not come to any disagreements during this deliverable.

Rebecca Di Filippo:

1. Creating a development plan before starting a project helps the team organize the scope, timeline, and responsibilities. It acts as a roadmap so we don't waste time moving in the wrong direction. The developmet plan also makes sure everyone is on the same page about attendance, meetings, roles and responsibilities. Without a plan, there is possibility of miscommunication, disorganization, and going in the wrong direction.

2. The biggest advantage of CI is that it automatically runs tests when you make a commit or open a pull request. That way, you know right when you commit if the code is safe to merge, which helps avoid integration problems and conflicts. CD makes deploying new versions faster and more reliable, which saves time and helps avoid mistakes. The downside to CI/CD is that it can take some effort, and it's annoying when builds are slow.

3. Our group didn't have major disagreements. The closest thing to a disagreement was, at first we thought CI/CD would be too difficult and unnecessary for the project. However, after discussing it with Dr. Istvan David, we concluded that it is worth including given the scope of our project. Its also a good learning experience.

Sunny Yao:

1. I have the opinion that initial planning is always highly important. It gets the team on the same page and sets expectations for the project. Most importantly, it helps set the vision for the project and define project scope. This lets members think about the technologies that will be used and the potential challenges.

2. An advantage of CI/CD is that it automates initial testing of future commits and pull requests. This helps catch bugs early and avoid builds crashing in production. The main disadvantage is the maintenance overhead from the setup as well as the time barrier needed for each pull request. This can then incentivize developers to make large infrequent commits to reduce the number of times that the CI/CD pipeline needs to be run.

3. We disagreed on initial project selection but each member was given a chance to make their case through votes and meeting discussions. This helped to reach a consensus on project direction. In addition, we were able to meet with our supervisor to get further motivations to pursue our final project idea.

Matthew Cheung:

1. Making a development plan is important because it acts as a roadmap for the entire project. It makes the team to sit down and explicitly think about the next steps we want to take to reach our goal. Without it, we would start coding without direction and end up wasting a ton of time developing the wrong things. The plan also helps us limit the scope of the project, setting distinct boundaries and limits to whatever work we do.

2. The biggest advantage for CI/CD is how it automates tedious tasks in the project. By automatically building and testing our code it saves the

team valuable time during development. This catches bugs earlier and ensures of no integration problems when we try to merge code, keeping the project stable. However, the downside is the overhead costs it takes to set up. It can be a lot of work to get the pipelines running, and for a small team like ours, the time could be costly. Additionally, if the builds are slow, it can become annoying and slow down our workflow.

3. We had some initial disagreements for our project choice, where we were debating between two different projects, but we ended up being force to choose Catan anyway due to prof availability so it didn't really matter. Once we got over the initial hurdle and started breaking down the work, we all got on the same page and were ready to work.

Appendix — Team Charter

External Goals

Our teams external goals for this project are to attain a strong understanding of reinforcement learning and computer vision, to be able to build a hireable portfolio. We also are aiming to get a A/A+ in the course, since we plan to put a lot of effort into the project.

Attendance

Expectations

Our team meets every week at 4:30-6:30 pm on Thursday. All other meetings are scheduled on a weekly basis, depending on deadlines and amount of work. It is expected that all team members attend every meeting and arrive on time. If a team member is unable to attend a meeting they must notify the team at least 12 hours in advance (withholding an emergency situation). If a team member is going to be late or must leave early they must notify the team prior to the meeting.

Acceptable Excuse

An acceptable excuse for missing a meeting or deadline includes sickness, family emergencies, or academic obligations. On the other hand, unacceptable excuses include forgetting, being lazy, or not prioritizing the team. Again, it is expected that the team knows 12 hours in advance if a member is going to miss a meeting or deadline. This time frame allows the team to adjust their plans accordingly.

In Case of Emergency

In case of an emergency, the team member must notify the team as soon as possible, before the meeting or deadline. If a deadline cannot be met, other

members will distribute the work among themselves to ensure the deadline is met. Its important to note that 1-2 missed deadline with little notice is acceptable, but repeated offenses will not be tolerated.

Accountability and Teamwork

Quality

It is expected that all team members come to meetings prepared with the assigned task completed for each week. It is expected that all tasks we complete are of high quality and have been reviewed by at least one other team member before submission. Each deliverable will be reviewed by all team members before submission to ensure quality and consistency. Lastly, all coding should follow the agreed upon coding standards.

Attitude

The team leader is responsible for ensuring that all team members are treated fairly. All team members need to be open to new ideas and willing to compromise. If conflicts arise, the team will discuss the issue and come to a resolution that works for everyone. If a team member is not contributing to the team, the team leader will address the issue with the member privately. If the issue persists, the team will discuss the issue with the TA or instructor.

Stay on Track

Our team will stay on track by setting clear weekly goals, using GitHub Kanban boards to monitor progress, and holding regular check-ins to ensure accountability. Tasks will be distributed fairly according to skills and proficiencies, and progress will be tracked through version control commits and documented updates. Members who perform well will be recognized for their contributions and can take greater lead on project direction in future tasks. If a member's performance falls below expectations, we will first address the issue through direct communication and support, ensuring that individual effort is accurately reflected in evaluations. We expect to have high attendance from all members with prior warning for any absences.

Team Building

Considering our team will be working together for 8 months, its important to build team cohesion. We will do this by scheduling fun activities outside of meetings. one of those fun activities will be a team dinner at the end of the project to celebrate. We also plan to play *Catan* together outside of meetings.

Decision Making

Our team will converse together before making any major decisions. We will try to reach a consensus, but if we are unable to do so we will vote through

discord polls. In the event there is a disagreement , the team leader will have the final say.