2   Project Plan

2.1 PROJECT MANAGEMENT/TRACKING PROCEDURES

Which of agile, waterfall or waterfall+agile project management style are you adopting? Justify it with respect to the project goals.

Waterfall planning with an agile development process. We have a defined plan for the project, but need to be flexible around the client.

What will your group use to track progress throughout the course of this and the next semester. This could include Git, Github, Trello, Slack or any other tools helpful in project management.

Gitlab and Trello

2.2 TASK DECOMPOSITION

In order to solve the problem at hand, it helps to decompose it into multiple tasks and subtasks and to understand interdependence among tasks. This step might be useful even if you adopt agile methodology. If you are agile, you can also provide a linear progression of completed requirements aligned with your sprints for the entire project.

 *Current list is not final

- Frontend development
  - Google Maps API integration
    - Different map overlays
  - UI features
    - Color coded timeline
    - Color coded map
    - Tabs/sections for graphs and overlays
  - Data visualization
    - Precipitation graphing
    - Graphing data in correct locations

- Backend development
  - Acquire data
    - Model with static data
    - Model with live data
  - Storing data
    - Connect server to database
  - Developing communication model with the frontend
    - Well-defined endpoints

■ Mapping frontend to backend

2.3 PROJECT PROPOSED MILESTONES, METRICS, AND EVALUATION CRITERIA

What are some key milestones in your proposed project? It may be helpful to develop these milestones for each task and subtask from 2.2. How do you measure progress on a given task? These metrics, preferably quantifiable, should be developed for each task. The milestones should be stated in terms of these metrics: Machine learning algorithm XYZ will classify with 80% accuracy; the pattern recognition logic on FPGA will recognize a pattern every 1 ms (at 1K patterns/sec throughput). ML accuracy target might go up to 90% from 80%.

In an agile development process, these milestones can be refined with successive iterations/sprints (perhaps a subset of your requirements applicable to those sprint).

- Working system with 1 watershed
- Prediction with static data
- Successful communication between frontend and backend
- A working use of Google Maps
- 5 second page response time

Tasks will be measured by client satisfaction with these specific predefined tasks.

2.4 PROJECT TIMELINE/SCHEDULE

• A realistic, well-planned schedule is an essential component of every well-planned project
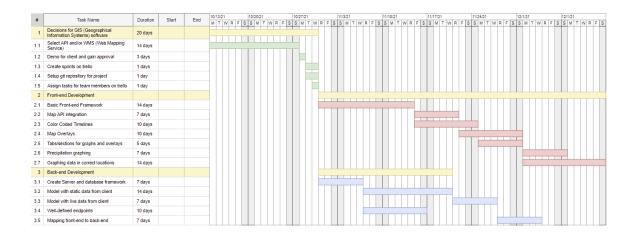
• Most scheduling errors occur as the result of either not properly identifying all the necessary activities (tasks and/or subtasks) or not properly estimating the amount of effort required to correctly complete the activity

• A detailed schedule is needed as a part of the plan:

– Start with a Gantt chart showing the tasks (that you developed in 2.2) and associated subtasks versus the proposed project calendar. The Gantt chart shall be referenced and summarized in the text.

– Annotate the Gantt chart with when each project deliverable will be delivered

• Project schedule/Gantt chart can be adapted to Agile or Waterfall development models. For agile, a sprint schedule with specific technical milestones/requirements/targets will work.

#	Task Name	Duration	Start	End
1	Decisions for GIS (Geographical Information Systems) software	20 days		
1.1	Select API and/or WMS (Web Mapping Service)	14 days		
1.2	Demo for client and gain approval	3 days		
1.3	Create sprints on trello	1 days		
1.4	Setup git repository for project	1 day		
1.5	Assign tasks for team members on trello	1 day		
2	Front-end Development			
2.1	Basic Front-end Framework	14 days		
2.2	Map API integration	7 days		
2.3	Color Coded Timelines	10 days		
2.4	Map Overlays	10 days		
2.5	Tabs/sections for graphs and overlays	5 days		
2.6	Precipitation graphing	7 days		
2.7	Graphing data in correct locations	14 days		
3	Back-end Development			
3.1	Create Server and database framework	7 days		
3.2	Model with static data from client	14 days		
3.3	Model with live data from client	7 days		
3.4	Well-defined endpoints	10 days		
3.5	Mapping front-end to back-end	7 days		

2.5 Risks And Risk Management/Mitigation

Consider for each task what risks exist (certain performance targets may not be met; certain tool may not work as expected) and assign an educated guess of probability for that risk. For any risk factor with a probability exceeding 0.5, develop a risk mitigation plan. Can you eliminate that task and add another task or set of tasks that might cost more? Can you buy something off-the-shelf from the market to achieve that functionality? Can you try an alternative tool, technology, algorithm, or board?

Agile projects can associate risks and risk mitigation with each sprint.

- Frontend development
  - Maps API integration - 0.5, we've never done maps API integration before. We can research alternatives and find the best one for our uses.
    - Different map overlays - 0.2
  - UI features
    - Color coded timeline - 0.1
    - Color coded map - 0.5, not a Google Maps API feature, we would have to find another way to integrate color coding into a map. If it does not work, then we will have to find a new maps API.
    - Tabs/sections for graphs and overlays - 0.05
  - Data visualization
    - Precipitation graphing - 0.2
    - Graphing data in correct locations - 0.2

- Backend development
  - Acquire data
    - Model with static data - 0.1
    - Model with live data - 0.4
  - Storing data
    - Connect server to database - 0.3
  - Developing communication model with the frontend
    - Well-defined endpoints - 0.1

- Mapping frontend to backend - 0.3

2.6 PERSONNEL EFFORT REQUIREMENTS

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be the projected effort in the total number of person-hours required to perform the task.

Maps API integration	20 hours
Different map overlays	4 hours
UI features	20 hours
Color coded timeline	3 hours
Color coded map	4 hours
Tabs/sections for graphs and overlays	2 hours
Data visualization	20 hours
Precipitation graphing	10 hours
Graphing data in correct locations	10 hours
Acquire data	15 hours
Model with static data	7.5 hours
Model with live data	7.5 hours
Storing data	8 hours
Connect server to database	8 hours
Developing communication model with the frontend	15 hours
Well-defined endpoints	5 hours
Mapping frontend to backend	10 hours

2.7 OTHER RESOURCE REQUIREMENTS

Identify the other resources aside from financial (such as parts and materials) required to complete the project.

We may need to help our client find meteorological data sources that are open and free to use for the final production system. All other resources are covered by our project plan.