# Rendering

1. Make the shaders.
2. Texture Loading
3. Frame buffer objects
4. Models for the Cars
5. MultiView Ports
6. GUI
   1. HUD during the racing
   2. Main menu display.
7. Modeling of the Maps
   1. Drawing the map object to the screen.
8. Power ups
   1. Drawing / Displaying the power-ups.

# AI

1. Controller Input
   1. Gathering input from the controller **(Milestone #1 – Jan.23 - Kent)**
   2. Providing an API / Integrating input into the rest of the program
      1. Direct API to use **(Milestone #2 – Jan 30. - Kent)**
      2. Use though event system **(Milestone #2 – Jan27- Kent)**
   3. Tuning input mapping to control the cars. **(Milestone # 2 - Feb 1. - Kent)**
2. AI Driving / Control of Cars
   1. Creating Framework to control cars (so that it is the same for human players and computer players) **(Milestone # 2 – Feb 5 - Kent)**
   2. Creating waypoint following / path finding for AI players.
      1. Maybe done by example, (person plays, AI tries to follow), or manual waypoints. **(Milestone #3 – Feb 17- Kent)**
      2. Possible ways of AI to cheat / make so the AI is actually a challenge **(Milestone #4 – Mar. 15- Kent)**
      3. Varying Difficulties **(Final – April 1- Kent)**
   3. AI collision detection / handling when AI gets stuck (off road etc.)
      1. Resetting the car when stuck. **(Milestone #3 – Mar 1. - Kent)**
3. Control of Camera behind the car.
   1. Implementation and integration of Camera into the game. **(Milestone 2 – Feb 10- Kent)**
   2. Introduction of slight delay / fixing camera “feel” of a “real” car. **(Milestone 3 – Feb 28- Kent)**
   3. Allow player to vary camera position slightly **(Milestone 3 – Feb 28- Kent)**
   4. Try to make a smart camera (rotates if needed, or makes objects transparent if needed) **(Final – April 5- Kent)**
4. Power-Ups
   1. Path finding when being shot. **(Milestone #3 – Mar 3. - Kent)**
5. Car Interplay with Dangerous Objects on the Track
   1. If car is damaged, etc. **(Milestone #4 – Mar. 20th – Kent)**

# Physics

1. Modeling of the maps
   1. Physics representation, and interaction.
2. Power Ups
   1. Collision detection on the power-ups
3. Spring Implementation
   1. Calculations Done
   2. Collision between springs and the floor (map)
      1. Detection if off ground
   3. Applied to the car (Holding up box)
   4. Tuning
4. Collisions
   1. Collision detection.
   2. Ability to set up collision boxes.
   3. Notifying the rest of the program / Integration
5. Integration with graphics / rendering
6. Ability to switch between simulation and faking physics

# Sound

1. Able to load and play sound files. (And acquiring sound files)
2. Integration into the game.
   1. General Game music playing (main menu, during game etc.)
   2. Car noises while driving (engine, braking)
   3. Specialized Sounds Occur when event occurs (e.g. crashing)

# Game play

1. Implementation of Game Rules
   1. Implementation of timing of each car.
   2. Keeping track of each lap and the number of laps completed.
   3. Number of power-ups for each car.
2. Designing Track
3. Tuning Vehicle mechanics.
4. Organizing Testing