Project Development of Sprint-1

TEAM ID	PNT2022TMID52163
PROJECT NAME	Signs with smart connectivity for better road safety

Sprint -1

```
import pygame
from pygame import gfxdraw
import numpy as np


class Window:
    def __init__(self, sim, config={}):
        # Simulation to draw
        self.sim = sim

        # Set default configurations
        self.set_default_config()

        # Update configurations
        for attr, val in config.items():
            setattr(self, attr, val)

    def set_default_config(self):
        """Set default configuration"""
        self.width = 1400
```

```
    self.height = 1000
    self.bg_color = (250, 250, 250)

    self.fps = 60
    self.zoom = 5
    self.offset = (0, 0)

    self.mouse_last = (0, 0)
    self.mouse_down = False



def loop(self, loop=None):
    """Shows a window visualizing the simulation and runs the loop function."""
    # Create a pygame window
    self.screen = pygame.display.set_mode((self.width, self.height))
    pygame.display.flip()

    # Fixed fps
    clock = pygame.time.Clock()

    # To draw text
    pygame.font.init()
    self.text_font = pygame.font.SysFont('Lucida Console', 16)

    # Draw loop
```

```
    running = True
    while not self.sim.stop_condition(self.sim) and running:
        # Update simulation
        if loop: loop(self.sim)

        # Draw simulation
        self.draw()

        # Update window
        pygame.display.update()
        clock.tick(self.fps)

        # Handle all events
        for event in pygame.event.get():
            # Handle mouse drag and wheel events
            ...



def convert(self, x, y=None):
    """Converts simulation coordinates to screen coordinates"""
    ...

def inverse_convert(self, x, y=None):
    """Converts screen coordinates to simulation coordinates"""
    ...
```

```
def background(self, r, g, b):
    """Fills screen with one color."""
    ...

def line(self, start_pos, end_pos, color):
    """Draws a line."""
    ...

def rect(self, pos, size, color):
    """Draws a rectangle."""
    ...

def box(self, pos, size, color):
    """Draws a rectangle."""
    ...

def circle(self, pos, radius, color, filled=True):
    """Draws a circle"""
    ...

def polygon(self, vertices, color, filled=True):
    """Draws a polygon"""
```

```
    def rotated_box(self, pos, size, angle=None, cos=None, sin=None, centered=True, color=(0, 0,
255), filled=True):
        """Draws a filled rectangle centered at *pos* with size *size* rotated anti-clockwise by *angle*.
"""

    def rotated_rect(self, pos, size, angle=None, cos=None, sin=None, centered=True, color=(0, 0,
255)):
        """Draws a rectangle centered at *pos* with size *size* rotated anti-clockwise by *angle*."""



    def draw_axes(self, color=(100, 100, 100)):
        """Draw x and y axis"""

    def draw_grid(self, unit=50, color=(150,150,150)):
        """Draws a grid"""

    def draw_roads(self):
        """Draws every road"""

    def draw_status(self):
        """Draws status text"""



    def draw(self):
        # Fill background
        self.background(*self.bg_color)
```

```
# Major and minor grid and axes
self.draw_grid(10, (220,220,220))
self.draw_grid(100, (200,200,200))
self.draw_axes()

# Draw roads
self.draw_roads()

# Draw status info
self.draw_status()
```