```
from crewai import Agent
from tools.PortfolioOptTools import PyPortfolioTools
from tools.MathTool import MathTool
from tools.ResearcherTools import ResearcherTools
from tools.MT5Tools import MT5Tools
from tools.TimeAndDateTool import TimeAndDateTool


class Gtraders:
    def __init__(self, model="ollama/phi3.5:3.8b"):
        self.model = model
        self.manager = self._create_manager()
        self.portfolio_manager = self._create_portfolio_manager()
        self.fundamental_analyst = self._create_fundamental_analyst()
        self.technical_analyst = self._create_technical_analyst()
        self.researcher = self._create_researcher()
        self.trader = self._create_trader()

    def _create_manager(self):
        return Agent(
            role="Manager",
            goal="Make high-level decisions and delegate tasks to other agents",
            backstory="An experienced financial manager with a track record of successful trading strategies and team
leadership.",
            verbose=True,
            allow_delegation=True,
            human_input=True,
            llm=self.model
        )

    def _create_portfolio_manager(self):
        return Agent(
            role="Portfolio Manager",
            goal="Optimize the investment portfolio for maximum returns while managing risk",
            backstory="A seasoned portfolio manager with expertise in asset allocation and risk management.",
            verbose=True,
            allow_delegation=True,  # Added delegation capability
            tools=PyPortfolioTools.get_tools(),
            llm=self.model
        )

    def _create_fundamental_analyst(self):
        return Agent(
            role="Fundamental Analyst",
            goal="Analyze economic indicators, financial reports, and market trends to provide insights for investment
decisions",
            backstory="An experienced financial analyst with a deep understanding of macroeconomic factors and their
impact on markets.",
            verbose=True,
            allow_delegation=True,  # Added delegation capability
            llm=self.model
        )

    def _create_technical_analyst(self):
        return Agent(
            role="Technical Analyst",
            goal="Perform technical analysis on forex, indices, and commodities to identify trading opportunities",
```

```
            backstory="A skilled technical analyst with expertise in chart patterns, indicators, and market trends.",
            verbose=True,
            allow_delegation=True,  # Added delegation capability
            tools=MathTool.get_tools(),
            llm=self.model
        )

    def _create_researcher(self):
        return Agent(
            role="Researcher",
            goal="Gather and analyze financial news, economic data, and market trends to support decision-making",
            backstory="A diligent researcher with a knack for uncovering valuable market insights and trends.",
            verbose=True,
            allow_delegation=True,
            tools=ResearcherTools.get_tools(),
            llm=self.model
        )

    def _create_trader(self):
        return Agent(
            role="Trader",
            goal="Execute trades based on analysis and market conditions while managing time-sensitive operations",
            backstory="An experienced trader with quick decision-making skills and a deep understanding of market
dynamics.",
            verbose=True,
            allow_delegation=True,  # Added delegation capability
            tools=MT5Tools.get_tools() + TimeAndDateTool.get_tools(),
            llm=self.model
        )

    def get_all_agents(self):
        return [
            self.manager,
            self.portfolio_manager,
            self.fundamental_analyst,
            self.technical_analyst,
            self.researcher,
            self.trader
        ]
```