To enable your AI model to ingest injury reports and \*\*accurately evaluate the impact of injured or absent players\*\* in college football, you’ll want to architect a pipeline that combines \*\*NLP, statistical analysis, and contextual modeling\*\*. Here’s a step-by-step breakdown of how you might build this, as well as important considerations for expansion:

---

## \*\*1. Data Ingestion\*\*

### \*\*A. Injury Reports\*\*

- \*\*Sources:\*\* Official NCAA feeds, team sites, Rotowire, ESPN, or CollegeFootballData API.

- \*\*Parsing:\*\* Use NLP (Named Entity Recognition) to extract:

- Player Name

- Team

- Position

- Injury status (out, questionable, probable, etc.)

- Game/Date

### \*\*B. Player Stats\*\*

- Gather season and historical (up to 5 years) stats per player, such as:

- Yards, TDs, tackles, etc.

- Snap counts, usage rates, advanced metrics (EPA, success rate)

### \*\*C. Opponent Defensive/Offensive Stats vs Position\*\*

- Aggregate how that week’s opponent performs \*\*against that position\*\*:

- E.g., “Yards allowed per game to RBs,” “Completion % allowed to WRs,” etc.

- Gather this for season & up to 5 years for context.

---

## \*\*2. Data Structuring\*\*

- Create relational tables or feature vectors:

- \*\*Player\*\*: {Name, Team, Position, Stat Profile, Injury Status}

- \*\*Opponent\*\*: {Team, Defensive/Offensive vs Position Stats, Conference}

- \*\*Historical Context\*\*: {Year, Game, Player, Performance, Opponent, Conference}

- \*\*Expandability:\*\* Structure data for easy grouping/analysis by player, team, conference, and opponent.

---

## \*\*3. Impact Evaluation Algorithm\*\*

### \*\*A. Baseline Player Value\*\*

- Estimate player’s “replacement value”:

- Compare injured player’s stats (per-game, per-snap) to team average, backup, or league average at position.

- Use advanced metrics (e.g., PFF grades, EPA added/lost, WAR if available).

### \*\*B. Opponent Context Adjustment\*\*

- Adjust expected impact based on opponent’s performance against similar players/positions.

- E.g., If a team is weak vs TE, missing a star TE is more significant.

- Use rolling averages, Z-scores, or percentiles to measure opponent strength vs position.

### \*\*C. Simulation/Scoring\*\*

- Simulate team performance with/without player using:

- Linear regression, ensemble models, or even simple “player out, drop X% of production” heuristics.

- Factor in historical matchups (last 5 years) for added context when data is sufficient.

---

## \*\*4. Output & Expansions\*\*

- \*\*By Player:\*\* Quantitative “impact score” or expected points differential.

- \*\*By Team:\*\* Aggregate all injuries to estimate overall team impact.

- \*\*By Conference/Opponent:\*\* Identify trends (e.g., SEC teams more resilient at RB).

- \*\*Explainability:\*\* Show which factors (player stats, opponent weakness) drive the impact score.

---

## \*\*5. Automation & Updates\*\*

- \*\*Automated Crons/Scripts:\*\* Pull new injury reports and stats weekly.

- \*\*Model Retraining:\*\* Update model weights as new seasons/data are available.

---

## \*\*6. Technologies to Use\*\*

- Pandas/SQL for data merging and grouping.

- Scikit-learn/XGBoost for modeling.

- SpaCy/NLTK for NLP on injury text.

- APIs: [CollegeFootballData API](https://collegefootballdata.com/api/docs), ESPN, Rotowire, etc.

---

## \*\*Example Pseudocode (Python-like)\*\*

````python

# Ingest injury report and stats

injuries = ingest\_injury\_reports() # NLP processing

player\_stats = get\_player\_stats(years=5)

opponent\_stats = get\_opponent\_vs\_position\_stats(years=5)

for injury in injuries:

player = injury['player']

position = injury['position']

team = injury['team']

opponent = get\_next\_opponent(team)

player\_value = calc\_player\_value(player\_stats[player])

opponent\_dvp = opponent\_stats[opponent][position]

backup\_value = get\_backup\_value(player\_stats, team, position)

impact\_score = player\_value - backup\_value

context\_adjusted\_impact = adjust\_for\_opponent(impact\_score, opponent\_dvp)

print(f"{player} ({team}) out vs {opponent}: Impact {context\_adjusted\_impact}")

````

---

## \*\*Further Expansions\*\*

- \*\*Cluster Analysis\*\*: Group similar injuries/impacts by conference, team, or player archetype.

- \*\*Historical Modeling\*\*: Use past games where key players were absent to refine impact estimates.

- \*\*Visualization\*\*: Heatmaps of “injury impact” by team/opponent, or time series for trends.

---

## \*\*Summary Table: Features to Implement\*\*

| Feature | Description / Expansion |

|-------------------------|----------------------------------------------|

| Injury NLP | Extract player, status, position |

| Player Stat Modeling | Baseline value, backup, team avg |

| Opponent DvP | Defensive/Offensive vs Position stats |

| Historical Context | Expand to 5 years, compare across teams/conf |

| Impact Scoring | Differential, context adjusted |

| Output Granularity | Player, Team, Conference, Opponent |

| Automation | Weekly updates, retraining |

---

\*\*Let me know if you want a full sample Python script, a database schema, or a system diagram for this approach!\*\*