```
@startuml

title Use Case Diagram - AI-Powered HR Recruitment System

actor "HR Manager" as HR

actor "Candidate" as C

actor "AI Engine" as AI

rectangle "AI-Powered HR Recruitment System" {

    (Login)

    (Submit Resume)

    (Parse Resume)

    (Analyze Skills)

    (Rank Candidates)

    (Schedule Interview)

    (Generate Report)

}

C --> (Submit Resume)

HR --> (Login)

HR --> (Rank Candidates)

HR --> (Schedule Interview)

HR --> (Generate Report)

(Submit Resume) .> (Parse Resume) : <<includes>>

(Parse Resume) .> (Analyze Skills) : <<includes>>

(Analyze Skills) .> (Rank Candidates) : <<includes>>

(Rank Candidates) .> (Generate Report) : <<extends>>

AI --> (Analyze Skills)

AI --> (Rank Candidates)
```
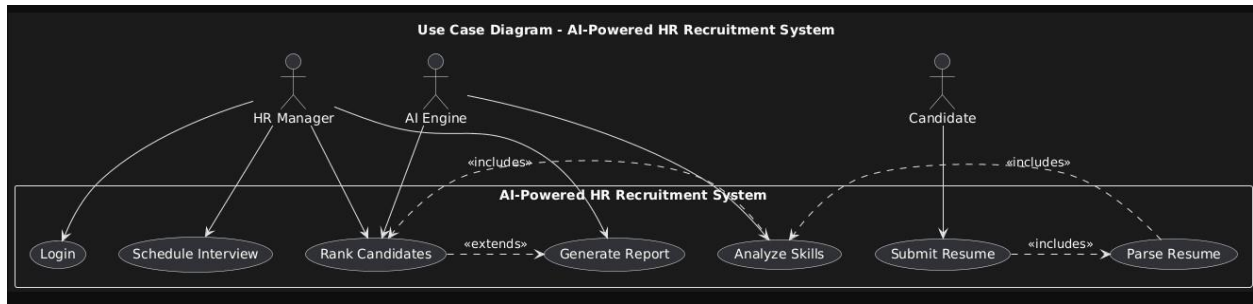
@enduml

Use Case Diagram - AI-Powered HR Recruitment System

@startuml

title Class Diagram - AI-Powered HR Recruitment System

class "User" {

  +userId: int

  +name: String

  +email: String

  +login(): void

}

class "Candidate" {

  +resume: File

  +skills: List

  +submitResume(): void

}

class "HRManager" {

  +reviewCandidates(): void

  +scheduleInterview(): void

  +generateReport(): void

}

```
class "ResumeParser" {

 +extractText(): String

 +extractSkills(): List

}


class "AIAnalyzer" {

 +evaluateSkills(candidateSkills: List): double

 +scoreCandidates(): void

}


class "RankingEngine" {

 +rankByScore(): void

 +generateFinalList(): List

}


class "ReportGenerator" {

 +generateReport(): void

 +exportReport(): void

}


User <|-- Candidate

User <|-- HRManager

Candidate --> ResumeParser : uses >

ResumeParser --> AIAnalyzer : sends skills >

AIAnalyzer --> RankingEngine : forwards scores >

RankingEngine --> ReportGenerator : outputs report >

HRManager --> ReportGenerator : accesses >

@enduml
```
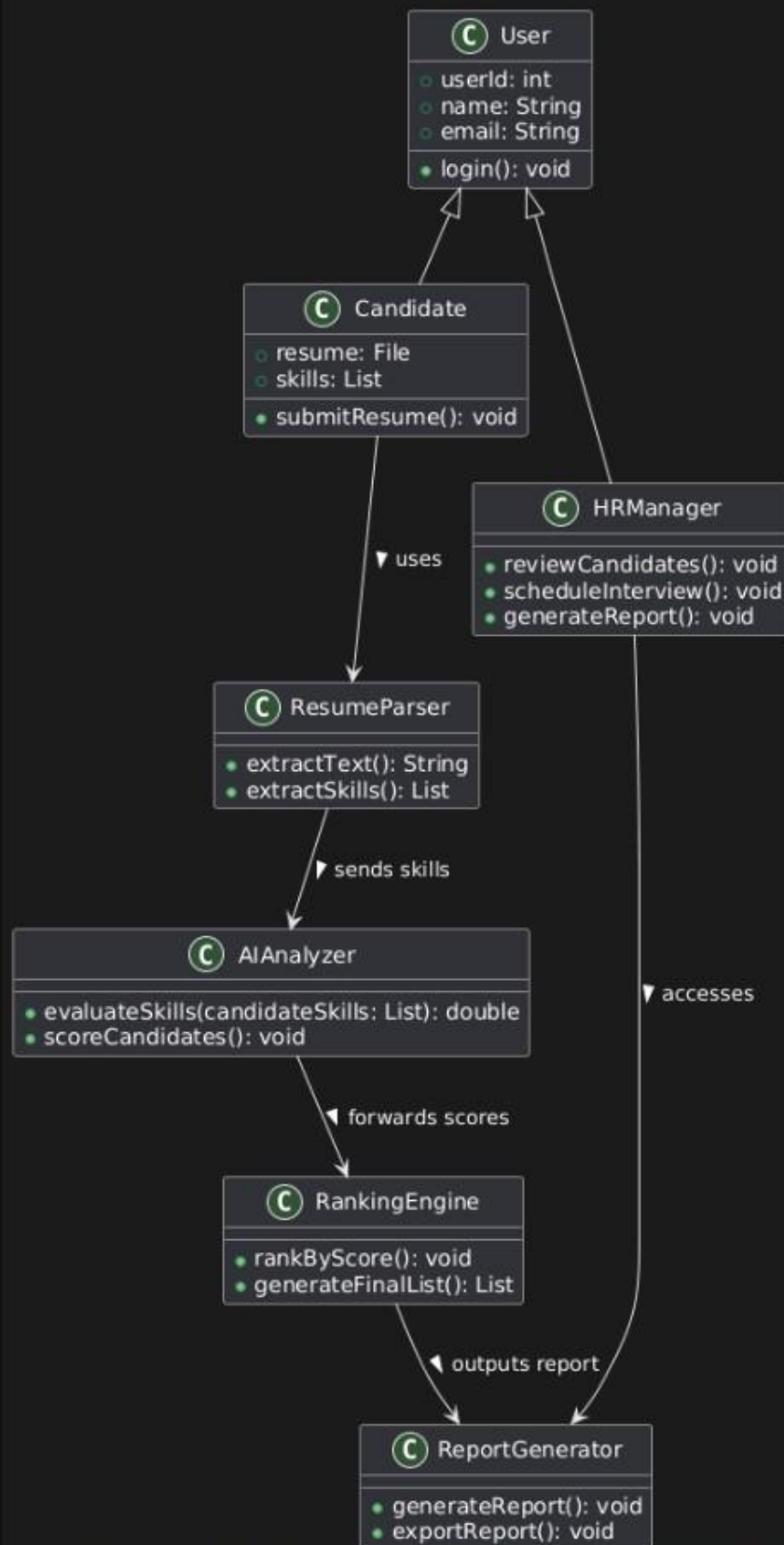
Class Diagram - AI-Powered HR Recruitment System

User
- userId: int
- name: String
- email: String
- login(): void

Candidate
- resume: File
- skills: List
- submitResume(): void

HRManager
- reviewCandidates(): void
- scheduleInterview(): void
- generateReport(): void

uses

ResumeParser
- extractText(): String
- extractSkills(): List

sends skills

AIAnalyzer
- evaluateSkills(candidateSkills: List): double
- scoreCandidates(): void

forwards scores

RankingEngine
- rankByScore(): void
- generateFinalList(): List

accesses

outputs report

ReportGenerator
- generateReport(): void
- exportReport(): void

```
@startuml

title Activity Diagram - AI-Powered HR Recruitment System


start

:Candidate submits Resume;

:Resume Uploaded;

:Parse Resume Data;

:Extract Skills using NLP;

:Analyze and Score Candidates;

if (Score above threshold?) then (Yes)

  :Add to Shortlist;

  :Notify HR Manager;

else (No)

  :Reject Candidate;

endif

:Generate Candidate Report;

:HR Schedules Interview;

stop


@enduml
```
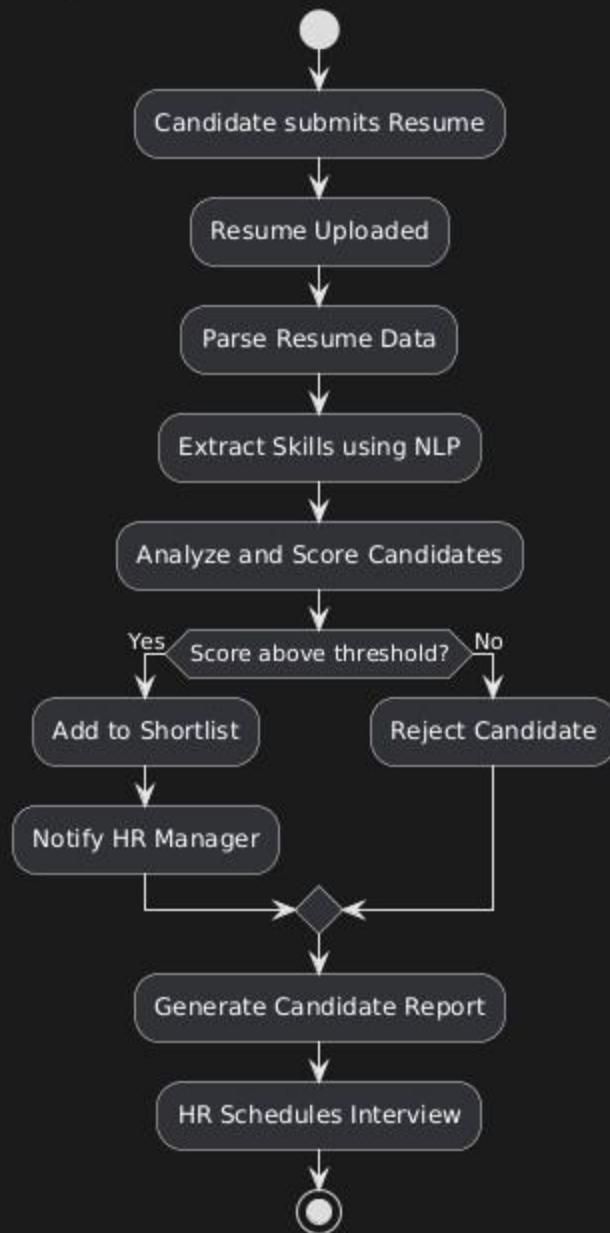
Activity Diagram - AI-Powered HR Recruitment System

@startuml

title Sequence Diagram - AI-Powered HR Recruitment System

actor Candidate

actor HRManager

participant "ResumeParser" as RP

participant "AIAnalyzer" as AI

participant "RankingEngine" as RE

participant "ReportGenerator" as RG

Candidate -> RP : Submit Resume

RP -> AI : Extracted Skills

AI -> AI : Analyze & Score Candidates

AI -> RE : Send Scores

RE -> RG : Generate Ranked Report

RG -> HRManager : Send Final Candidate List

HRManager -> Candidate : Notify Shortlisted

@enduml