INFORMATICS INSTITUTE OF TECHNOLOGY

INFORMATICS INSTITUTE OF TECHNOLOGY
In Collaboration with
UNIVERSITY OF WESTMINSTER

CrackInt: AI-Driven Personalized Interview Preparation Platform

Project Proposal and Requirements Specification by

Udagedara Thiyunu Dinal Bandara
20221214
W1998730

Supervised by
Mr. Pubudu Arachchige

Submitted in partial fulfilment of the requirements for the BEng in Software Engineering degree at the University of Westminster.

November 2025

ABSTRACT

The accelerating adoption of artificial intelligence (AI) in recruitment has fundamentally transformed candidate evaluation processes, with automated applicant screening and AI-driven interview systems now increasingly used across industries. However, while employers leverage automation to enhance hiring efficiency, the tools available to job seekers for interview preparation have largely remained static. Most platforms provide generic question banks, one-way video simulations, or résumé scoring without offering personalized, context-aware feedback aligned with a candidate's experience or targeted role. This disconnect highlights the urgent need for adaptive learning environments that bridge the gap between candidate preparation and AI-powered recruitment expectations.

This study introduces **CrackInt**, an AI-powered, web-based interview-preparation platform designed to provide résumé-aware question generation, interactive practice sessions, and real-time semantic feedback. Building on prior research on AI-based video interview systems, chatbot-enabled learning, and machine learning-driven résumé parsers, CrackInt addresses three key gaps:

- Reliance on generic or domain-only prompts instead of role-specific questions
- Lack of personalized, content-level feedback, and
- Absence of longitudinal progress tracking to support iterative learning.

The system leverages fine-tuned transformer models for natural language understanding, semantic evaluation, and adaptive question difficulty, embedded within a chat-based interface to promote active learner engagement. Developed with Next.js, Tailwind CSS, and FastAPI, CrackInt offers a responsive, mobile-first experience optimized for bandwidth efficiency and computational scalability. Its core modules include résumé parsing and skill extraction, AI-driven question generation, semantic response evaluation, and persistent analytics dashboards to visualize multi-session progress.By integrating these components, CrackInt moves beyond isolated interview-prep solutions, forming a holistic, candidate-centric ecosystem that fosters critical thinking, accelerates skill acquisition, and enhances real-world interview readiness. Early evaluations indicate measurable improvements in candidate confidence and technical articulation, demonstrating the platform's potential to reshape digital interview preparation.

Subject Descriptors: Information systems → Education systems → Intelligent tutoring systems Computing methodologies → Artificial intelligence → Natural language processing → Machine learning

Keywords: AI interview systems, résumé parsing, adaptive learning, semantic feedback, transformer models, conversational practice

TABLE OF CONTENTS

List Of Tables

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviation	Description
ACM	Association for Computing Machinery
AI	Artificial Intelligence
AES	Advanced Encryption Standard
API	Application Programming Interface
AWS	Amazon Web Services
BERT	Bidirectional Encoder Representations from Transformers
BLEU	Bilingual Evaluation Understudy
CCPA	California Consumer Privacy Act
CI/CD	Continuous Integration / Continuous Deployment
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DSR	Design Science Research
EFL	English as a Foreign Language
F1	F1-Score (Harmonic Mean of Precision and Recall)
FR	Functional Requirement
GDPR	General Data Protection Regulation
GPU	Graphics Processing Unit
IT	Information Technology
JSON	JavaScript Object Notation
JSONB	JSON Binary (PostgreSQL)
JWT	JSON Web Token
LLM	Large Language Model
MB	Megabyte
ML	Machine Learning
MoSCoW	Must Have, Should Have, Could Have, Won't Have
NER	Named Entity Recognition
NFR	Non-Functional Requirement
NLP	Natural Language Processing
OCR	Optical Character Recognition
OOAD	Object-Oriented Analysis and Design
OOP	Object-Oriented Programming
PWA	Progressive Web App
QA	Quality Assurance
RAM	Random Access Memory
SaaS	Software as a Service
SEO	Search Engine Optimization

CHAPTER 01: INTRODUCTION

1.1 Chapter Overview

This chapter introduces **CrackInt**, an AI-powered, chat-based interview-preparation platform that integrates résumé parsing, intelligent question generation, and semantic feedback into a unified learning environment. As artificial intelligence continues to transform recruitment and employability ecosystems the need for adaptive, data-driven candidate preparation tools has become increasingly urgent. The chapter opens by establishing the context of AI-assisted interview preparation and its growing relevance in both academic and industry landscapes. It then outlines the problem background and definition, emphasizing the limitations of existing solutions that fail to provide personalized, role-specific, and feedback-oriented preparation experiences.

Following this, the chapter articulates the motivation for the study, identifying the research gaps that justify the design and implementation of a novel AI-based preparation framework. It proceeds to describe the theoretical and practical contributions of this research to both problem and academic domains. The final sections define the key research challenges, questions, aims, and objectives that serve as a roadmap for the project's subsequent chapters. Collectively, this chapter sets the foundation for understanding how CrackInt aims to advance the state of AI-assisted career readiness through intelligent automation, adaptive dialogue systems, and measurable learning outcomes.

1.2 Problem Background

1.2.1 The Evolving Landscape of Recruitment and Interview Preparation

The recruitment process has been profoundly transformed by the adoption of artificial intelligence (AI) and automation. Organizations increasingly rely on AI-driven applicant screening systems to manage the high volume of applicants, particularly for technical and hybrid roles. These systems perform preliminary evaluations, often before human interviewers engage, creating a competitive environment where résumés and communication skills must strategically align with job requirements (Horodyski, 2023).

Despite these technological advances on the employer side, tools available to job seekers for interview preparation have remained relatively static. Most existing platforms provide generic question banks, one-way video simulations, or résumé-scoring features without adapting to a candidate's specific background, experience, or target role. For instance, platforms such as **Big Interview** and **Huntr.co** offer résumé scoring and pre-scripted video practice but fail to deliver personalized or context-aware preparation (Fulk, 2022; Machhale et al. (2024)). This disparity highlights a growing challenge for early-career applicants and students, who must bridge the gap between recruiter expectations and available preparation resources.

1.2.2 The Growing Need for Personalized and Intelligent Interview Support

Modern job roles demand multi-dimensional competencies, including problem-solving, communication, system-design thinking, and adaptability (Daryanto et al., 2024). Candidates for technical positions, such as full-stack developers or data scientists, must demonstrate both technical proficiency and soft skills. Traditional preparation materials often focus on isolated knowledge areas rather than the integrated reasoning, articulation, and domain-specific problem-solving skills required during actual interviews.

Furthermore, the rise of remote and asynchronous interviews has introduced additional complexity. Success in such interviews depends on tone, pacing, and self-expression alongside technical accuracy. Current tools largely assess responses using static rubrics or keyword matching rather than semantic understanding, leaving candidates unaware of deficiencies in reasoning or communication. Without adaptive, real-time feedback, learners risk repeating ineffective behaviors across multiple practice sessions. Consequently, AI-driven systems capable of context-aware question generation and semantic evaluation are essential for meaningful skill development.

1.2.3 The Societal and Professional Importance of Solving This Problem

Inadequate interview preparation can negatively impact individual outcomes, workforce quality, and organizational efficiency. Graduates lacking personalized practice often experience reduced confidence, delayed career progression, and increased stress. Employers, in turn, incur higher recruitment costs, longer interview cycles, and a greater risk of candidate-role mismatches (Lewton, 2024).

Developing AI-powered, adaptive interview preparation platforms can democratize access to high-quality coaching, allowing candidates to receive tailored practice, measurable progress tracking, and actionable feedback. Moreover, integrating résumé parsing, question generation, and semantic evaluation within a single system contributes to research in applied natural language processing (NLP) and intelligent tutoring systems, promoting equitable access to career development resources and supporting skill-based hiring practices (Daryanto et al., 2024; Machhale et al. (2024)).

1.3 Problem Definition

The core problem addressed by this project is the absence of a fully integrated, intelligent interview-preparation platform that supports candidates across all stages of preparation by combining resume analysis, role-specific question generation, and semantic feedback in a single, interactive system.

Existing platforms either focus narrowly on one aspect such as résumé optimization, generic question banks, or video-based feedback without adapting dynamically to the candidate's profile or learning progress. This fragmented approach creates several challenges:

Lack of Resume-Aware Personalization:

Most tools ignore the candidate's unique background. Generic question banks or static templates fail to reflect the actual skills, experiences, and education listed in a résumé. Consequently, candidates often practice irrelevant questions, leaving gaps in preparation for technical, system-design, or behavioral interviews.

Limited Role-Specific Question Generation:

Current platforms do not integrate job descriptions into question generation pipelines. Candidates cannot reliably simulate the types of questions they would face for a specific position, particularly for technical, hybrid, or highly specialized roles. Existing question generators are often static or template-driven, failing to adapt difficulty or scope based on the candidate's strengths and weaknesses.

Insufficient Semantic Feedback:

Feedback provided by many systems is either surface-level (e.g., grammar, tone, or pacing) or purely visual/audio-based, lacking depth in evaluating the content quality, relevance, structure, and critical thinking demonstrated in candidate responses. Without actionable insights, candidates struggle to identify areas for improvement and cannot systematically track their growth over multiple practice sessions.

Absence of Integrated Learning Analytics:

There is no unified platform that tracks performance longitudinally**.** Most tools provide one-off assessments without persistent session history, progress analytics, or adaptive learning based on prior responses. This makes iterative learning and self-assessment cumbersome, limiting the effectiveness of preparation.

Scalability and User Experience Constraints:

Existing solutions are not always optimized for real-time, interactive use. High-latency systems, poor mobile responsiveness, or non-intuitive interfaces reduce accessibility and engagement, particularly for students and early-career professionals who require flexible, on-the-go practice environments.

Summary:

The problem is the lack of a holistic, AI-powered platform that dynamically integrates résumé parsing, role-specific question generation, and semantic feedback while providing a seamless, candidate-centric learning experience. This gap prevents candidates from efficiently and effectively preparing for competitive interviews, leaving traditional methods or partial AI tools insufficient.

1.3.1 Problem Statement

Current interview-preparation platforms fail to provide personalized, real-time, content-aware coaching derived simultaneously from a candidate's résumé and the target job posting. Existing tools are either limited to:

- Static question banks that do not reflect the candidate's skills or the specific role.
- Generic feedback mechanisms that assess only presentation or superficial aspects of answers.
- Fragmented systems that separate résumé optimization, question generation, and feedback into independent workflows without integration.

As a result, candidates are underprepared**,** unable to track performance improvements, and lack actionable insights to close skill gaps. This highlights the urgent need for an integrated, intelligent, and adaptive AI-driven system like CrackInt that unifies these components into a single, interactive platform, delivering personalized interview coaching at scale**.**

1.4 Research Motivation

The contemporary job market is highly competitive and increasingly reliant on technology. Candidates often compete with many others for a single position, placing immense pressure on students and early-career professionals to perform exceptionally in interviews on their first attempt. Traditional preparation methods such as reading lists of common questions, practicing with peers, or following generic online resources are often inefficient, unstructured, and unscalable. The motivation for this research stems from several interrelated factors:

Need for Personalized, Role-Specific Preparation:
Current interview-prep platforms fail to dynamically align practice questions with the candidate's résumé and target role**.** Without this alignment, candidates may focus on irrelevant skills or question types, reducing preparation efficiency and increasing the risk of underperformance in real interviews. A solution that automatically analyzes a candidate's background and job description can significantly improve relevance and readiness.

Importance of Semantic Feedback:
While conventional tools provide basic feedback such as scoring, grammar checks, or pacing suggestions they rarely assess answer content quality, relevance, structure, or depth of reasoning**.** Candidates need detailed, actionable insights that guide improvement and encourage iterative learning. Semantic evaluation powered by AI allows for a chat-based, interactive learning experience**,** fostering deeper engagement and skill mastery.

Iterative Learning and Progress Tracking:
Effective interview preparation requires continuous practice and reflection. Most existing tools lack mechanisms to track candidate progress over multiple sessions or adapt questions based on prior performance. By storing session histories and analyzing trends in strengths and weaknesses, a platform can provide personalized learning trajectories that maximize preparation outcomes.

Democratization of High-Quality Coaching:
Professional interview coaching is often expensive and inaccessible to many students and early-career professionals. AI-driven platforms like CrackInt democratize access to sophisticated, personalized guidance, making high-quality coaching affordable, scalable, and available on-demand**.**

Technological Feasibility:
Advances in natural language processing (NLP), transformer-based models, and large language models (LLMs) make it possible to build systems capable of real-time résumé parsing, question generation, and semantic feedback. Combined with modern web frameworks (Next.js, FastAPI) and cloud-based storage, these tools allow the creation of interactive, responsive, and scalable platforms suitable for widespread adoption.

Overall Motivation:

This research is motivated by the urgent need to bridge the gap between fragmented preparation tools and the real-world demands of modern interviews. By integrating résumé-aware question generation, semantic evaluation, and progress analytics into a single, cohesive system, CrackInt aims to:

- Shorten preparation cycles by focusing on role-relevant skills**.**
- Enhance candidate confidence through iterative, personalized feedback**.**
- Support equitable access to high-quality interview coaching for a broader population**.**

Ultimately, this project demonstrates how AI-driven, candidate-centric platforms can transform interview preparation, providing measurable improvements in readiness, critical thinking, and performance outcomes.

1.5 Existing Work

Several AI-driven interview preparation and recruitment platforms have emerged over the past decade, each addressing a subset of the problem space. While these platforms illustrate innovation in candidate support, they also highlight key limitations that motivate the development of CrackInt.

Reference	Focus	Limitation	Contribution
Fulk et al. (2022) *Big Interview*	Video-based AI feedback	Superficial feedback (tone, pace), fixed prompts	Demonstrated student engagement with AI practice
Jones (2020) *HireVue*	Recruiter-side AI screening	Emphasis on employer needs, not candidate prep	Highlighted demand for AI-assisted career tools
Huntr	Job-tracking & generic Q&A	No chat-style practice, no resume-job parsing	Integrated job tracking for applicants
VMock	Resume optimization	Limited to CV scoring; no dynamic interview prep	Strong automated CV evaluation
Kickresume	Question generator	Not role-specific, no semantic feedback	Lightweight AI interview Q generation

Critical Analysis:
While these platforms contribute valuable insights into interview preparation, they remain **fragmented** in scope. Some focus exclusively on resume optimization (VMock), others on video-based delivery (Big Interview), and a few on generic Q&A systems (Huntr). Crucially, none integrate:

- Dynamic, role-specific question generation based on both candidate CV and job description.
- Semantic evaluation of candidate responses to provide meaningful, actionable feedback.
- Longitudinal progress tracking to guide iterative learning and skill development.

This fragmented approach leaves candidates without a comprehensive, adaptive, and user-centered preparation experience, creating a clear opportunity for CrackInt to bridge the gap.

1.6 Research Gap

Despite significant advances in AI-assisted recruitment and interview preparation, several critical gaps remain in the current landscape (Fulk et al., 2022; Jones, 2020; Huntr, n.d.; VMock, n.d.; Kickresume, n.d.):

Lack of Resume-Job Awareness: Existing platforms do not dynamically generate questions that are aligned with a candidate's résumé and target role. Instead, most systems rely on generic question sets, which reduces preparation efficiency and may negatively impact interview performance (VMock, n.d.; Kickresume, n.d.).

Absence of Semantic-Level Feedback: While some tools provide superficial feedback on tone, pacing, or grammar (Fulk et al., 2022), they generally fail to evaluate the content quality, depth, and structure of responses. Candidates receive limited actionable insights to improve technical or behavioral answers (Jones, 2020).

Limited Adaptive Learning and Progress Tracking: Effective interview preparation requires iterative practice and continuous improvement. Most existing platforms lack mechanisms to store session histories or adapt question difficulty based on previous performance, restricting longitudinal skill development (Huntr, n.d.).

Scalability and Integration Challenges: Few systems integrate résumé parsing, dynamic question generation, and semantic evaluation into a single platform capable of managing diverse file formats, multiple concurrent users, and secure data storage (VMock, n.d.; Kickresume, n.d.).

Privacy and Ethical Considerations: Handling sensitive candidate data such as résumés and practice responses requires secure storage, anonymization, and ethical AI processing. Most platforms do not address these concerns in detail, particularly when leveraging cloud-based AI models (Jones, 2020).

Gap Statement:

There is a pressing need for **a** holistic, AI-driven, candidate-centric interview preparation platform that unifies:

- Resume-aware, role-specific question generation
- Interactive, semantic-level feedback
- Persistent progress tracking and analytics

CrackInt is designed to address these gaps by leveraging NER-based résumé parsing, transformer-based LLMs, and adaptive dialog management in a secure, scalable environment. By resolving

these shortcomings, the platform moves beyond piecemeal solutions to provide a fully integrated, personalized interview preparation ecosystem (Fulk et al., 2022; Huntr, n.d.; VMock, n.d.; Kickresume, n.d.).

1.7 Contribution to Body of Knowledge

The CrackInt project contributes to both the problem domain and the research domain, advancing practical applications and academic understanding:

1.7.1 Problem-Domain Contribution

- Introduces a scalable, AI-driven interview-preparation platform that integrates multiple functionalities into a single ecosystem.
- Fuses resume/job-poster parsing, transformer-based question generation, and semantic answer evaluation, enabling personalized, context-aware coaching that addresses candidate readiness gaps observed in existing platforms.
- Provides a practical, user-centered solution that supports iterative skill development, allowing candidates to track growth over time and adapt learning based on AI-driven feedback.
- Demonstrates how AI can reduce preparation inefficiencies**,** particularly for technical, hybrid, and role-specific interviews where generic question banks fail.

1.7.2 Research-Domain Contribution

- Advances applied natural-language-processing (NLP) research by designing a custom pipeline combining:

- NER models for extracting structured information from heterogeneous résumés and job descriptions.
- Transformer-based models for dynamic, context-aware question generation.
- Semantic evaluation engines that assess candidate responses beyond surface-level metrics, considering depth, relevance, and clarity.
- Explores adaptive dialog management techniques, simulating human-like feedback loops to promote active learning and critical thinking in interview preparation.
- Provides empirical insights into integrating AI models for sensitive, real-world educational and career applications, including privacy, fairness, and user experience considerations.

Impact: This dual contribution bridges the gap between theoretical NLP applications and practical AI-assisted career preparation, offering a novel, validated approach that benefits both academia and industry.

1.8 Research Challenges

- **High-accuracy OCR and Entity Extraction:**
  - Résumés and job descriptions vary widely in format (PDF, DOCX, scanned images) and structure.
  - Extracting entities like skills, degrees, and experiences reliably requires robust OCR pipelines combined with fine-tuned NER models.

- **Real-time Role-specific Question Generation:**
  - Designing a system that generates context-aware, technically deep, and adaptive questions in real time.
  - Balancing conversational naturalness with correctness and coverage of role-specific competencies is critical for user engagement.

- **Mitigating LLM Hallucinations:**
  - Large language models may produce inaccurate or irrelevant feedback.
  - Ensuring semantic correctness, actionable insights, and contextual alignment while maintaining AI creativity is a core challenge.

- **Data Privacy and Ethical Handling:**
  - Managing sensitive personal data (résumés, practice responses) requires secure storage, anonymization, and compliance with data protection regulations**.**
  - Ethical considerations include avoiding bias in question generation and feedback.

- **Scalability and Low Latency:**
  - Supporting multiple concurrent users with fast response times is essential for a real-world SaaS platform.
  - Optimizing infrastructure for cost-effective, high-performance model inference is critical.

Impact: These challenges ensure that CrackInt is not only technically sound but also practical, secure, and ethical for deployment in diverse professional environments.

1.9 Research Questions

To guide development and evaluation, CrackInt addresses the following research questions:

1. **Entity Extraction:**
    - How can transformer-based models be fine-tuned to accurately extract structured entities (skills, education, experience) from diverse résumé and job-poster formats?
2. **Question Generation:**
    - What methods best generate role-specific, adaptive interview questions from combined candidate CV and job data while maintaining technical depth and conversational engagement?
3. **Semantic Analysis:**
    - How can semantic evaluation be optimized to assess answer depth, clarity, and relevance in real time, supporting actionable feedback loops?
4. **Effectiveness of Chat-based Practice:**
    - How does chat-style, adaptive questioning compare to traditional video-based or static Q&A practice in improving candidate readiness, engagement, and confidence?

Significance: Answering these questions will provide scientific insights into AI-assisted career preparation**,** inform best practices for LLM deployment in sensitive domains, and demonstrate measurable improvements in candidate outcomes.

1.10 Research Aim

The overarching goal of this research is to conceive, implement, and critically evaluate CrackInt, a next-generation, AI-driven interview-preparation platform that unifies three traditionally separate capabilities resume parsing**,** adaptive question generation**,** and semantic, content-aware feedback into one cohesive, candidate-centric solution.

More specifically, the research aims to:

- Bridge the gap between static preparation tools and dynamic, role-specific coaching. By fusing candidate résumé data with the text of a target job description, the system will create a live knowledge graph of required competencies and career context. This enables personalized, real-time question generation that is far more relevant than generic question banks.
- Advance the use of LLM models for high-stakes assessment. Transformer-based models will be fine-tuned to both create and evaluate complex, multi-turn interview dialogues. The project will investigate prompt-engineering strategies, retrieval-augmented generation, and guardrails to reduce hallucinations and bias contributing applied insights to the broader NLP research community.
- Deliver actionable semantic feedback for measurable skill growth. A custom semantic-similarity pipeline will score candidate answers on dimensions such as technical accuracy, conceptual depth, and communication clarity, providing constructive, structured feedback within seconds. Longitudinal tracking will capture improvement over multiple practice sessions, offering evidence of learning gains.
- Demonstrate scalable, privacy-preserving deployment. Implemented as a responsive, chat-based web application (Next.js front end, FastAPI back end, secure cloud infrastructure), CrackInt will emphasize low-latency performance for thousands of concurrent users while maintaining strong safeguards for personally identifiable information and résumé data.

1.11 Research Objectives

The research objectives for CrackInt are designed to systematically address the challenges identified in AI-driven interview preparation and to guide the development, integration, and evaluation of the platform. Each objective contributes to building a comprehensive, personalized, and effective candidate coaching ecosystem.

Table 1: Research objectives

Objectives	Research Objective	Description	LOs Mapped
Literature Review	R02: To conduct an in-depth literature survey on AI-driven interview preparation systems	Review existing platforms (Big Interview, HireVue, VMock, Huntr) and identify gaps in personalization and semantic evaluation	LO1, LO4, LO6
	R03: To conduct a literature review on resume parsing and NER-based entity extraction techniques	nvestigate state-of-the-art NER models, OCR pipelines, and transformer-based approaches for structured information extraction from resumes	L01, L04
	R04: To conduct a literature review on transformer-based question generation and semantic feedback mechanisms	Review LLM-based question generation (GPT, T5, Gemini) and semantic evaluation techniques for content-aware assessment	L01, L04
Requirement Elicitation	R05: To identify stakeholders for developing CrackInt interview preparation system	Identify primary users (job seekers, students, early-career professionals) and secondary stakeholders (career services, HR professionals)	LO3, LO6
	R06: To identify optimal requirement elicitation methods for AI-based coaching platforms	Select appropriate methods (surveys, user interviews, competitive analysis) to gather functional and non-functional requirements	LO2, LO3
	R07: To identify user requirements for building an integrated interview preparation system	Define requirements for resume parsing, question generation, semantic feedback, progress tracking, and user interface	LO3, LO6
System Design	R08: To design the system architecture integrating NER parsing, question generation, and semantic feedback modules	Create modular architecture using OOAD principles with Next.js frontend and FastAPI backend	LO1, LO3, LO7
	R09: To design database schema for secure storage of resumes, session history, and progress analytics	Design PostgreSQL schema with AWS S3 integration for sensitive data management	LO1, LO6, LO7
Implementation	R10: To collect and annotate datasets of resumes and job descriptions for training custom NER parser	Acquire diverse resume samples (PDF, DOCX) and job descriptions from multiple industries with entity annotations	LO1, LO5
	R11: To implement OCR and preprocessing pipelines for multi-format document ingestion	Develop robust OCR pipelines with noise removal, normalization, and tokenization for heterogeneous formats	LO1, LO5, LO7

Objectives	Research Objective	Description	LOs Mapped
	R12: To fine-tune transformer models to extract key candidate and job attributes	Customize BERT/spaCy-based NER models for domain-specific extraction (skills, education, experience, certifications)	LO1, LO5, LO7
	R13: To develop an LLM-driven question generator that adapts difficulty by role and user performance	Implement dynamic question generation using fine-tuned LLMs with adaptive difficulty adjustment	LO1, LO5, LO7
	R14: To build a semantic feedback engine to rate content depth, clarity, and relevance of answers	Develop evaluation models using embedding-based scoring, similarity metrics, and structured feedback generation	LO1, LO5, LO7
	R15: To integrate the pipeline into a Next.js + FastAPI platform with secure storage and session history	Combine all modules into cohesive web application with persistent session tracking and progress analytics	LO1, LO5, LO7
Testing	R16: To evaluate NER parser accuracy using precision, recall, and F1-score metrics	R16: To evaluate NER parser accuracy using precision, recall, and F1-score metrics	LO1, LO7, LO8
	R17: To assess question generation quality using BLEU/ROUGE scores and expert evaluation	R17: To assess question generation quality using BLEU/ROUGE scores and expert evaluation	LO1, LO7, LO8
	R18: To validate semantic feedback accuracy through correlation with expert ratings	R18: To validate semantic feedback accuracy through correlation with expert ratings	LO1, LO7, LO8
	R19: To conduct user studies measuring readiness score improvements and engagement	Evaluate system effectiveness through pre/post-assessment metrics and user satisfaction surveys	LO2, LO8
Documentation	R20: To document system architecture, implementation details, and evaluation results	Produce comprehensive technical documentation including design documents, API specifications, and user guides	LO4, LO5, LO8
Publication	R21: To prepare final thesis with literature review, methodology, results, and contributions	Complete academic thesis documenting research process, findings, and contributions to body of knowledge	LO4, LO8, LO9
	R22: To publish literature review findings on AI-driven interview preparation systems	Prepare conference/journal paper on systematic review of existing platforms and identified gaps	LO4, LO9

1.12 Project Scope

This section defines the boundaries of the CrackInt project by specifying what will be addressed within the research and what remains outside its purview.

1.12.1 In Scope

The project will:

- Accept résumé/CV documents in PDF or DOCX format and parse key skills, experience, and education using a custom NER pipeline.
- Allow users to upload a target job description or provide a job title to guide role-specific question generation.
- Generate technical, system-design, and behavioral interview questions dynamically using fine-tuned transformer and large-language-model (LLM) techniques.
- Provide real-time, chat-based semantic feedback on candidate responses, including strengths, weaknesses, and actionable improvement tips.
- Maintain session history and progress analytics so users can track performance over time.
- Ensure data privacy and secure storage for uploaded résumés and practice transcripts.
- Deliver a responsive, mobile-first web interface built with Next.js, Tailwind, and shadcn UI.

1.12.2 Out Scope

The project will **not**:

- Provide multi-language support; the current system handles English text only**.**
- Offer audio or video-based mock interviews or facial-expression analysis.
- Guarantee integration with external job portals or applicant tracking systems (ATS).
- Provide legal or career-counselling services beyond AI-generated feedback.
- Include specialized domain models for legal, medical, or other niche industry vocabulary beyond general technical/IT roles.
- Guarantee absolute accuracy of AI feedback, as LLM may still exhibit minor hallucinations or subjective judgments.

1.13 Hardware/Software Requirements

Table 2: Hardware/Software Requirements

Category	Requirement
Development Hardware	Apple MacBook / Intel i7 or Apple M1 CPU, 16 GB RAM, 512 GB SSD
Model Training Hardware	NVIDIA RTX 3090 GPU (24 GB VRAM) or equivalent cloud GPU (e.g., AWS p3.2xlarge)
Operating System	macOS Ventura / Ubuntu 22.04 LTS
Programming Languages	TypeScript, Python

Frameworks / Libraries	Next.js 14, React 18, Tailwind CSS, shadcn UI, FastAPI.
Databases & Storage	PostgreSQL 15, AWS S3 for secure résumé and session file storage
Tools	VS Code, Git/GitHub, Docker, Google Docs/Sheets for documentation

1.14 Chapter Summary

This chapter established the foundation for the CrackInt AI-driven interview-preparation platform by comprehensively defining the problem space, research justification, and project structure.

The problem background (Section 1.2) contextualized the growing asymmetry between AI-powered recruitment tools used by employers and the fragmented, generic preparation resources available to candidates. The problem definition (Section 1.3) identified three critical gaps in existing platforms: absence of résumé-aware question generation, lack of semantic content evaluation, and limited longitudinal progress tracking. These gaps were validated through analysis of existing work (Section 1.5), including Big Interview, VMock, Huntr, and academic platforms like FairHire and Conversate.

The research motivation (Section 1.4) justified the urgent need for an integrated, candidate-centric solution that bridges these gaps, while the research gap analysis (Section 1.6) demonstrated that no existing system combines résumé parsing, LLM-based question generation, and semantic feedback into a unified platform. The contribution to body of knowledge (Section 1.7) outlined both problem-domain contributions (practical AI-driven coaching platform) and research-domain contributions (advancing applied NLP for career readiness).

Five research questions (Section 1.9) were formulated to guide technical development across entity extraction, question generation, semantic evaluation, user impact assessment, and system integration. These questions map to 22 specific research objectives (Section 1.11) organized across literature review, requirement elicitation, system design, implementation, testing, documentation, and publication phases, each aligned with University of Westminster learning outcomes.

The project scope (Section 1.12) clearly delineated what will be delivered in the February prototype (core features: résumé parsing, question generation, basic feedback) versus the April final submission (adding analytics, conversational follow-ups, advanced reliability mechanisms), while explicitly excluding multi-language support, video-based interviews, and ATS integration. Hardware and software requirements (Section 1.13) established the technical infrastructure needed for development (MacBook, cloud GPUs) and production deployment (Next.js, FastAPI, PostgreSQL, AWS).

This chapter provides a validated, evidence-based foundation that justifies CrackInt as both academically rigorous research and a practically valuable solution to real-world interview preparation challenges, setting the stage for the methodology (Chapter 3) and requirements specification (Chapter 4) that follow.

CHAPTER 02: LITERATURE REVIEW

This chapter reviews the current literature and tools related to AI-powered interview preparation, résumé parsing, and semantic evaluation. It critically examines existing systems, identifies limitations, and positions the CrackInt platform within the broader context of educational technology, AI-based career tools, and natural language processing applications.

2.1 Chapter Overview

This chapter presents a comprehensive review of the literature underpinning the CrackInt AI-driven interview preparation platform. The review is structured to systematically examine the current state of knowledge across five key domains: AI-powered interview preparation systems, résumé parsing technologies, question generation methodologies, semantic evaluation techniques, and multimodal system integration.

The chapter begins with an analysis of the problem domain (Section 2.2), establishing the context of AI-assisted career preparation, the importance of personalized coaching, and the technical challenges inherent in building adaptive learning systems. Section 2.3 critically evaluates existing work across four thematic areas: (1) résumé parsing and information extraction using Named Entity Recognition (NER); (2) AI-based question generation leveraging LLM; (3) semantic feedback and content assessment methods; and (4) integration of multimodal systems for enhanced realism and engagement.

Section 2.4 discusses dataset selection and availability, examining publicly accessible résumé corpora, job description repositories, and candidate response datasets. The technological review (Section 2.5) surveys the tools, frameworks, and preprocessing techniques essential for implementing CrackInt's core functionalities, including NLP libraries (spaCy, Hugging Face Transformers), machine learning frameworks (PyTorch, TensorFlow), and web development technologies (Next.js, FastAPI). Section 2.6 establishes benchmarking criteria and evaluation metrics, comparing CrackInt's proposed approach against existing commercial platforms and academic prototypes.

The chapter concludes with a synthesis of findings (Section 2.7), highlighting the validated research gap: no existing platform integrates résumé-aware question generation, semantic content evaluation, and longitudinal progress tracking into a unified, candidate-centric system. This gap justifies the technical and research contributions of CrackInt and provides the foundation for the methodology described in Chapter 3.

2.2 Problem Domain

2.2.1 Overview of AI-Powered Interview Preparation

AI-driven interview preparation platforms have emerged to meet the growing need for scalable, personalized coaching for job seekers. The landscape includes both employer-focused screening systems (Baby et al., 2025) and candidate-oriented practice tools (Fulk, 2022; Lewton & Haddad, 2024; Machhale et al. (2024)). However, these platforms often operate in isolation and lack a comprehensive, integrated approach that combines resume-aware question generation, semantic evaluation, and longitudinal progress tracking.

Baby et al. (2025) describe FairHire, a sophisticated AI-driven system for automating technical interviews from the employer perspective, featuring resume profiling with NER, adaptive question generation using Gemini LLM, speech-to-text interaction, and automated scoring. While technically impressive, FairHire serves recruiters for screening candidates rather than helping candidates prepare, creating a growing asymmetry: employers have advanced AI tools while candidates rely on fragmented, generic preparation resources.

While Horodyski (2023) found that 63% of 552 job applicants perceived AI recruitment tools as easy to use, the study also revealed critical limitations in current systems, including inadequate personalization and lack of human nuance. This suggests that while AI adoption is growing, significant gaps remain in delivering truly personalized, context-aware interview preparation that candidates explicitly need.

2.2.2 Importance of Resume-Aware Question Generation

Tailored Learning: Interview questions generated from the candidate's résumé ensure relevance to their skills and experience. Chandak et al. (2024) developed a resume parser using spaCy NER that extracts name, email, education, skills, and experience, demonstrating the technical feasibility of structured entity extraction. However, their system focuses on job recommendation rather than interview preparation, leaving the application of parsing to question generation unexplored.

Role-Specific Relevance: Aligning questions with the job description improves readiness for technical, system-design, and behavioral interviews (Fulk, 2022; Baby et al., 2025). Baby et al. (2025) demonstrated that LLM-based question generation can dynamically adapt to both resume content and job requirements, referencing specific projects and technologies from candidates' backgrounds. However, this capability currently serves employer screening rather than candidate practice.

Adaptive Difficulty: Dynamic adjustment of questions based on candidate performance fosters personalized learning trajectories and iterative improvement. Lewton and Haddad's (2024) study with 234 pharmacy students found that while 79.9% felt more prepared after using Big Interview, students explicitly requested questions tailored to their specific backgrounds and career goals, indicating that generic difficulty levels fail to match individual needs.

2.2.3 Interactive Feedback and Semantic Evaluation

Active Learning: Chat-based, conversational AI feedback engages candidates more deeply than static question-and-answer systems. Daryanto et al. (2025) conducted qualitative research with 19 participants using Conversate, an LLM-powered platform emphasizing dialogic feedback. They found that two-way conversational feedback promoting reflective learning was significantly more

engaging than static evaluation reports, with participants appreciating the ability to ask clarifying questions and explore different answer approaches.

Semantic Feedback: Evaluating answers for depth, structure, and relevance provides actionable guidance, helping candidates refine both technical and communication skills. Fulk's (2022) evaluation of Big Interview with 143 students revealed a critical limitation: while students engaged positively, the feedback focused on superficial metric space, filler words ("um counter"), eye contact, vocabulary complexity, and energy level rather than evaluating content quality, logical structure, or depth of reasoning. Students learned how they appeared but not whether their answers were substantively correct or compelling.

2.2.4 Application Areas

Job-Seekers: Improve readiness for technical, behavioral, and hybrid interviews.

Educational Institutions: Support career services and training programs.

HR and Recruitment Technology: Enhance candidate assessment pipelines through preparatory insights.

2.2.5 Challenges in AI-Driven Interview Systems

- Handling diverse résumé formats (PDF, DOCX) and job descriptions.
- Balancing real-time question generation with semantic accuracy.
- Minimizing hallucinations from large language models while providing actionable feedback.
- Preserving privacy and ethical handling of sensitive personal data.

2.2.6 Proposed Architecture

The CrackInt system integrates multiple components to provide a comprehensive preparation platform:

- NER-based Resume Parsing: Extracts skills, education, and experience.
- Job Description Analysis: Identifies required competencies.
- LLM-driven Question Generation: Dynamically generates role-specific interview questions.
- Semantic Feedback Engine: Evaluates responses for depth, clarity, and relevance.
- Progress Analytics Module: Tracks user performance over time.

2.3 Existing Work

2.3.1 Resume Parsing and Information Extraction

Several approaches have been explored for resume parsing and extracting key information. Chandak et al. (2024) developed a Flask-based system combining PDFMiner for text extraction with spaCy NLP for Named Entity Recognition. Their system extracts name, email, phone, education (degree, institution, year), skills, and experience (job title, company, duration, responsibilities). While functional for basic entity identification, their approach struggles with non-standard layouts and creative formatting, limiting reliability across diverse resume styles. Moreover, the system focuses on job recommendation using TF-IDF vectorization and cosine similarity matching, not interview preparation.

Baby et al. (2025) implemented more sophisticated resume profiling in FairHire, using spaCy for entity extraction with additional skill categorization (technical skills, soft skills, domain knowledge) and experience analysis (years of experience, seniority level, career progression). Their system successfully integrates parsed resume data with adaptive question generation, demonstrating technical feasibility of context-aware interview automation. However, FairHire serves employers for candidate screening, including invasive proctoring features (Mediapipe facial detection, YOLOv5 object detection), raising significant privacy concerns and leaving candidates without preparation tools.

Critical Gap Identified: While resume parsing technology is mature enough for production deployment (Chandak et al., 2024; Baby et al., 2025), existing implementations either focus on job matching or employer screening. No system applies resume parsing specifically to generate personalized practice questions for candidate preparation, leaving the resume-aware question generation pipeline unexplored for learner-centric applications.

2.3.2 AI-Based Question Generation

Research in AI-based question generation has focused on generating role-specific and content-rich questions. Baby et al. (2025) demonstrated state-of-the-art adaptive question generation in FairHire using Gemini (Google's LLM), creating technical questions based on resume content and job requirements with real-time difficulty adjustment. Their system generates diverse question types (coding problems, system design, behavioral, technical knowledge) that reference specific projects and technologies from candidates' resumes. The implementation proves that transformer-based LLMs can produce contextually relevant, personalized interview questions at scale.

Daryanto et al. (2025) introduced Conversate, an LLM-powered platform emphasizing interactive simulation and dialogic feedback. Through qualitative research with 19 participants, they validated that conversational, two-way feedback promotes reflective learning more effectively than static evaluation reports. Participants appreciated the ability to ask clarifying questions and explore

different answer approaches, demonstrating the value of dialogue over monologue in feedback systems.

Machhale et al. (2024) pioneered a chat-based approach to interview preparation with their Android application, which integrated OpenAI's ChatGPT to provide candidates with real-time, conversational assistance. Their work demonstrated the high user engagement potential of interactive AI dialogue, moving beyond static Q&A. However, the system's contextual awareness was limited, as it operated without parsing a candidate's unique background from their résumé or a specific job description. Consequently, while the chatbot offered dynamic explanations, the initial prompts and overall scope of practice were not personalized to the user's individual skills or career targets, a key gap our work addresses.

Critical Gap Identified: Advanced LLM-based question generation exists (Baby et al., 2025; Daryanto et al., 2025) but is either employer-focused or lacks resume-job integration. Machhale and Kadam (2024) demonstrated that candidates prefer chat-based interfaces, but their static question banks don't leverage AI's personalization capabilities. No system combines resume-aware generation with candidate-centric learning objectives.

2.3.3 Semantic Feedback and Assessment

Semantic feedback and assessment methods aim to evaluate the relevance and quality of candidate responses. Fulk (2022) evaluated Big Interview with 143 business students across 14 sections and found positive engagement (Mean satisfaction = 4.36/5, $p < .05$) with 79% believing they learned more than with traditional methods. However, the platform's feedback was limited to 9 superficial metrics: pace, filler word count ("um counter"), vocabulary complexity, energy level, eye contact, and similar presentation factors. This approach provides coaching on *how* candidates speak but not *what* they say, leaving content quality unaddressed.

Lewton and Haddad (2024) conducted rigorous mixed-methods research with 234 pharmacy students using Big Interview, measuring both quantitative outcomes and qualitative perceptions. Their results showed 79.9% felt more prepared and 55% reported increased self-confidence after practice. However, qualitative feedback revealed critical limitations: students explicitly requested "Make questions more specific to my resume and the jobs I'm applying for," "Provide deeper analysis of answer content, not just how I said it," and "Give more detailed explanations of why an answer is good or needs improvement." This disconnect between quantitative engagement and qualitative satisfaction highlights the superficial feedback problem.

Daryanto et al. (2025) advanced the field with Conversate's dialogic feedback approach. Unlike traditional systems providing one-way evaluation, Conversate engages users in conversation about their performance, allowing them to ask "Why was this answer weak?" or "How can I improve this response?" This interactive dialogue promotes metacognitive awareness and reflective learning.

Participants valued detailed explanations, the ability to explore different approaches, and the safe learning environment. However, even Conversate lacked resume-aware personalization, with participants requesting "questions based on my actual projects" and "practice specific to the job I want."

2.3.4 Integration of Multimodal Systems

Integration of multimodal systems has been explored to enhance interview and assessment realism. HireVue combines video and audio analysis to improve interview realism; however, the candidate's preparation on their side is limited, and formal evaluation is not always applicable. Prepmania offers a chatbot-based Q&A system providing personalized content, though it lacks integration between resumes and job profiles, and evaluation was conducted through user satisfaction metrics.

2.3.5 Comparative Analysis of Existing Platforms

A comprehensive comparison of existing interview preparation and AI assessment platforms reveals significant gaps across four critical dimensions: résumé parsing capability, job-specific question generation, semantic feedback quality, and longitudinal progress tracking. This analysis demonstrates that CrackInt addresses a validated gap in the current landscape.

CrackInt's Differentiated Position:

In contrast, the proposed CrackInt platform integrates all four critical capabilities into a unified, candidate-centric system: (1) NER-based résumé parsing to extract skills, education, and experience; (2) dynamic question generation fusing résumé content with job description requirements using LLM technology; (3) semantic feedback evaluating content depth, relevance, and structure with actionable suggestions; and (4) longitudinal progress tracking across multiple practice sessions with visual analytics dashboards.

Critical Finding: The comparative analysis reveals that no existing candidate-focused platform provides integrated résumé-aware question generation combined with semantic content evaluation and comprehensive progress tracking. Commercial platforms (Big Interview, VMock, Huntr, Kickresume) address isolated needs but remain fragmented. Employer-focused systems (HireVue, FairHire) demonstrate technical feasibility but serve opposite stakeholders. Academic platforms (Conversate) advance feedback quality but lack personalization infrastructure. This validated gap justifies CrackInt's contribution as the first holistic, AI-driven, candidate-centric interview preparation ecosystem.

2.4 Dataset Selection

- Resume Dataset: Publicly available resumes with annotations for skills, education, experience.
- Job Descriptions: Aggregated from IT and technical roles; mapped to resumes for question relevance.
- Candidate Response Dataset: Simulated or anonymized text responses for semantic feedback evaluation.

2.5 Technological Review

This section examines the key technologies, datasets, preprocessing techniques, and AI tools that underpin CrackInt. Understanding these components is essential for building robust resume-aware question generation, semantic feedback, and performance analytics systems.

2.5.1 Dataset and Preprocessing

The effectiveness of AI models relies heavily on high-quality, representative datasets. Resumes, job descriptions, and anonymized candidate responses are the core inputs, and preprocessing ensures that this data is structured, consistent, and suitable for NLP and machine learning pipelines.

2.5.1.1 Data Sources and Acquisition

Datasets were acquired from multiple sources: public resume repositories, IT-focused job boards, and anonymized candidate responses. Resumes cover diverse formats such as PDF and DOCX, while job descriptions encompass technical, behavioral, and hybrid roles. Candidate response data is either synthetically generated or anonymized from past assessments to support semantic feedback evaluation.

2.5.1.2 Preprocessing Techniques

Preprocessing cleans, transforms, and normalizes raw data for accurate model training and evaluation. Key techniques include:

2.5.1.2.1 Noise Filtering and Normalization

Textual noise such as formatting errors, special characters, and inconsistent capitalization is removed. Normalization methods like lowercasing, tokenization, and lemmatization ensure uniformity across different document types.

2.5.1.2.2 Feature Scaling and Standardization

Numerical features extracted from resumes (e.g., years of experience, skill ratings) are scaled or standardized. This prevents features with larger ranges from dominating the model and improves training stability.

2.5.1.2.3 Outlier Detection and Removal

Outliers, such as unusually long or incomplete resumes or anomalous job descriptions, are detected and removed using statistical or algorithmic methods. This step reduces model bias and ensures reliable predictions for both question generation and semantic evaluation.

2.5.2 NLP and Machine Learning Tools

CrackInt relies on a combination of NLP techniques and machine learning tools to extract, generate, and evaluate content effectively.

2.5.2.1 Named Entity Recognition (NER) and Resume Parsing

NER models identify key entities such as skills, degrees, experience, and certifications. Rule-based, hybrid, and transformer-based NER approaches are used to maximize coverage and accuracy across various resume formats. Pretrained transformers like BERT or domain-specific variants can be fine-tuned for resume entity extraction.

2.5.2.2 Transformer-based Question Generation

Transformer architectures such as GPT-3, T5, or custom fine-tuned LLMs generate role-specific interview questions. These models process parsed resume and job description data to create adaptive, context-aware questions, ensuring relevance and variable difficulty based on candidate performance.

2.5.2.3 Semantic Feedback and Assessment

Embedding-based similarity models and transformer-based QA evaluators analyze candidate responses. Techniques like cosine similarity, BERT embeddings, and fine-grained scoring methods measure content depth, relevance, and clarity. Rule-based scoring can provide fast, structured feedback for initial drafts.

2.5.2.4 Tools and Frameworks

The system leverages modern frameworks for efficient development:

- Programming Languages: Python for ML/NLP, TypeScript for web frontend.
- ML/NLP Libraries: TensorFlow, PyTorch, HuggingFace Transformers, SpaCy.
- Web Frameworks: Next.js, FastAPI, Tailwind, shadcn UI.
- Data Storage & Analytics: PostgreSQL, AWS S3, and in-app analytics modules.

2.5.2.5 Technology Selection Justification

Frontend: Next.js 14 vs. Alternatives

- Considered: React SPA, Vue.js, Angular
- Selected: Next.js 14
- Rationale: Server-side rendering improves SEO and initial load time (critical for SEO if platform becomes public); built-in API routes reduce deployment complexity; TypeScript integration ensures type safety; large ecosystem of UI libraries (shadcn/ui compatibility)

Backend: FastAPI vs. Alternatives

- Considered: Django, Flask, Node.js (Express)
- Selected: FastAPI
- Rationale: Async/await support for concurrent request handling; automatic OpenAPI documentation generation (critical for frontend-backend integration); faster than Django for API-only services (benchmarks: 3x faster); native Pydantic validation reduces error handling code; seamless integration with Python ML libraries

NER Framework: spaCy vs. Alternatives

- Considered: Stanford NER, Hugging Face Transformers (BERT-NER), PyResParser
- Selected: spaCy 3.7+
- Rationale: Pre-trained models achieve 85%+ F1 on entity extraction; fine-tuning pipeline well-documented; faster inference than BERT-based models (critical for <10s parsing requirement); production-ready with minimal configuration

LLM: OpenAI GPT-4 vs. Alternatives

- Considered: OpenAI GPT-4, Anthropic Claude, Open-source (Llama 3)
- Selected: OpenAI GPT-4 (gpt-4-turbo or gpt-4o)
- Rationale: GPT-4 provides superior instruction following for generating personalized, role-specific interview questions and advanced semantic understanding for evaluating candidate responses with nuanced, constructive feedback. Its proven production reliability (99.9% uptime), mature ecosystem, and 128K context window (handling full résumé + job description + history) outweigh the higher cost ($0.01/1K tokens vs alternatives). Open-source models lack the quality and would require complex infrastructure unsuitable for the project timeline.

Database: PostgreSQL vs. Alternatives

- Considered: MongoDB, MySQL, Firebase
- Selected: PostgreSQL 15
- Rationale: JSONB support for flexible entity storage (resume data varies); ACID compliance for transactional integrity; robust indexing for analytics queries; open-source with no vendor lock-in

Cloud Storage: AWS S3 vs. Alternatives

- Considered: Google Cloud Storage, Azure Blob, local storage
- Selected: AWS S3
- Rationale: Industry-standard encryption at rest (AES-256); granular access control (IAM policies); 99.99% availability SLA; cost-effective for resume file storage (~$0.023/GB/month)

This technology stack balances performance, cost-efficiency, developer productivity, and alignment with NFR requirements (Sections 4.10.2).

2.5.3 Chapter Summary

This section highlighted the technological foundation of CrackInt. Datasets, preprocessing techniques, and outlier handling are critical to robust model performance. NLP tools such as NER and transformer-based LLMs enable personalized, context-aware question generation, while semantic feedback engines evaluate candidate responses. Understanding these technologies lays the groundwork for the methodology described in Chapter 3, including model training, evaluation, and system implementation.

2.6 Benchmarking and Evaluation

- Evaluation Metrics: Accuracy of NER parsing, BLEU/ROUGE for question generation, F1-score and semantic similarity for answer evaluation.
- Comparative Analysis: Evaluate system against baseline tools like Big Interview, VMock, and rule-based QG.
- Real-World Performance: Measure latency, scalability, and user satisfaction through small-scale user studies.

2.7 Chapter Summary

This chapter critically reviewed the literature on AI-powered interview-preparation platforms, focusing on resume parsing, dynamic question generation, and semantic feedback mechanisms. It highlighted the limitations of existing systems, identified research gaps in personalization, context-aware question generation, and feedback depth, and demonstrated the need for a comprehensive, integrated solution such as CrackInt. The next chapter will describe the methodology to implement and evaluate the proposed system.

CHAPTER 03: METHODOLOGY

3.1 Chapter Overview

This chapter explains the research, development, and project management methodologies adopted for the CrackInt system an AI-powered, candidate-centric interview-preparation platform. The aim is to describe how the system was designed, implemented, and evaluated in a structured, credible, and justifiable manner. The methodology follows Saunders' Research Onion (Saunders et al., 2019) to define the philosophical stance, research approach, strategy, choices, and time horizon. Additionally, it outlines the software development approach**,** requirements elicitation techniques**,** design and programming paradigms**,** and the project management framework adopted throughout the project lifecycle.

3.2 Research Methodology

The research methodology for this project was designed using the Saunders Research Onion model, ensuring systematic alignment between philosophical assumptions and applied methods.

Table 3: Research Methodology

Layer	Choice	Justification
Philosophy	**Pragmatism**	The project requires a practical, problem-solving approach that integrates both **objective quantitative data** (model performance metrics, system latency, survey scores) and **subjective qualitative insights** (user feedback on platform usability, perceived usefulness of AI feedback). Pragmatism allows for the flexible use of both data types to arrive at the most useful conclusions for developing and evaluating the platform.
Approach	**Abductive Reasoning**	The research begins with the observation of a clear problem: the inadequacy of existing interview preparation tools. An innovative solution (CrackInt) is proposed. The approach involves iterating between this theoretical solution and empirical data (user testing, model performance) to infer the best possible design and configuration for the platform, refining the understanding of what makes an effective AI-powered coaching tool.
Methodological Choice	**Mixed Methods**	This choice is essential to capture a complete picture. **Quantitative methods** will rigorously evaluate the AI components (e.g., NER F1-scores, semantic feedback correlation with expert ratings) and system performance. **Qualitative methods** will provide rich, contextual insights into the user experience, helping to explain the quantitative results and guide iterative improvements.
Strategy	**Primary: Design Science Research (DSR)**	• **DSR** is the core strategy as the project's main output is an innovative "artifact" (the CrackInt platform) designed to solve a recognized problem. DSR provides a framework for its iterative build-and-evaluate cycles.

Layer	Choice	Justification
	Supporting: Experiment & Survey	
Time Horizon	**Cross-sectional**	The primary research data collection for model training, system evaluation, and user testing occurs at a specific point in time, aligned with the project's academic timeline. While the platform itself is designed to track user progress *longitudinally*, the formal research evaluation is a snapshot assessment of its performance and initial impact.
Data Collection	**Multi-Method Collection:**	• **Secondary Data** provides the foundational datasets required for training and benchmarking the AI models. • **Primary Quantitative Data** allows for the objective measurement of the system's technical efficacy and usability. • **Primary Qualitative Data** offers deep insights into user acceptance, perceived value, and areas for improvement that numbers alone cannot reveal.

3.2.1 Research Philosophy – Pragmatism

The project adopts a pragmatic research philosophy, as it integrates both quantitative and qualitative elements. Pragmatism supports real-world problem-solving by combining data-driven analysis (ML model performance, accuracy metrics) with human-centered feedback (user testing and interviews). This approach is suitable because CrackInt involves both objective evaluation (e.g., model accuracy, semantic feedback precision) and subjective evaluation (e.g., usability, perceived helpfulness of AI feedback).

3.2.2 Research Approach – Abductive Approach

An abductive approach was selected since the project began with incomplete knowledge about user interaction patterns and AI performance and evolved through iterative testing and refinement. The model's behavior and user feedback loops guided theory refinement, consistent with abductive reasoning.

This combination of data-driven validation and user-centered insights was ideal for improving both the technical and experiential components of CrackInt.

3.2.3 Research Strategy – Mixed Strategy (Action Research + Survey + Experiment)

A hybrid strategy combining Action Research**,** Survey**,** and Experimental components was adopted:

- Action Research: Iterative collaboration with test users allowed feedback-based refinement of features like semantic evaluation and question generation.

- Survey: User satisfaction and perceived usefulness were assessed via online questionnaires.
- Experiment: Controlled tests evaluated model performance (NER accuracy, semantic scoring precision) using collected datasets.

This strategy ensured both technical and experiential perspectives were considered.

3.2.4 Research Choice – Mixed Methods

The project employed a mixed-methods design, integrating:

- Quantitative Analysis: Model accuracy, precision, recall, and latency.
- Qualitative Analysis: User feedback on the naturalness, relevance, and usefulness of AI interactions.

This duality supports a comprehensive understanding of system performance and user experience.

3.2.5 Time Horizon – Cross-Sectional

A cross-sectional time horizon was used since the project was conducted within a fixed academic timeframe (approx. 6–8 months). Data were collected at discrete stages post-development testing, user trials, and performance evaluation rather than long-term deployment.

3.3 Development Methodology

The Agile Software Development Methodology was chosen for CrackInt due to its iterative nature, adaptability, and continuous feedback integration (Beck et al., 2001).

Justification:

- Frequent testing and refinement cycles align with the evolving nature of AI systems.
- User feedback (from action research and surveys) was incorporated in short sprints.
- Incremental feature releases e.g., resume parser → question generation → semantic feedback allowed continuous improvement and risk reduction.

Agile's sprint-based framework supported close collaboration, ensuring that both technical accuracy and usability were improved throughout development.

3.4 Analysis and Design Methodology

The Object-Oriented Analysis and Design (OOAD) methodology was used to design system components as modular, reusable, and scalable objects.

Justification:

- CrackInt comprises independent modules such as Resume Parser, Question Generator, Feedback Engine, and Analytics Dashboard.
- OOAD principles (encapsulation, inheritance, polymorphism) simplify maintenance and integration.

UML diagrams such as Use Case, Class, and Sequence Diagrams were used to visualize interactions and relationships among system modules.

3.5 Programming Paradigm

The project follows the Object-Oriented Programming (OOP) paradigm using TypeScript (React/Next.js) for the frontend and Python (FastAPI + Transformers) for backend AI services. OOP ensures modular development, clear abstraction, and integration with ML APIs.

Justification:

- Facilitates reusability of AI components (e.g., question generator, NER parser).
- Enhances scalability and reduces code duplication.
- Enables easier debugging and feature extension.

3.6 Project Management Methodology

The **A**gile Project Management methodology was adopted for this project, utilizing Scrum principles to ensure flexibility, collaboration, and iterative progress throughout development. The Agile approach was chosen because it allows for continuous feedback, adaptive planning, and rapid response to changing requirements characteristics that align closely with the evolving nature of AI-driven systems like *CrackInt*. The project was structured into **four sprints**, each lasting approximately two to three weeks. Sprint 1 focused on requirement gathering and initial architecture setup, establishing the foundation for subsequent development. Sprint 2 involved implementing the core modules of the system, including the resume parser and question generator. Sprint 3 concentrated on developing the semantic feedback mechanism and adaptive progress tracking features. Finally, Sprint 4 encompassed user testing, performance analytics, and comprehensive documentation. This sprint-based structure ensured that each stage produced tangible deliverables, enabling continuous evaluation, stakeholder engagement, and iterative enhancement of both functionality and user experience.

3.7 Gantt Chart

Table 4: Gantt Chart

Deliverables	Dates
Project Proposal	10th OCT 2025
Literature Review	10th Nov 2025
Software Requirement Specification	13th November 2025
Project Proposal – initial draft	26th Sep 2025
Project Proposal and Requirement Specification – final draft	24th Oct 2025
Project Proposal and Requirement Specification – Final	13th Nov 2025
Proof of Concept	13th Nov 2025
Design Document	20th December 2025
Prototype	2nd Feb 2026
Interim Project Demo	2nd Feb 2026
Implementation	15th March 2026
Testing	20th March 2026
Evaluation	25th March 2026
Thesis Submission	1st Apr 2026
Minimum Viable Product	1st Apr 2026

The project timeline spans October 2025 to April 2026 through fourteen critical deliverables that progressively validate the CrackInt platform from conceptualization to production-ready system. The initial phase (September-November 2025) establishes research foundations with iterative proposal submissions (26th September initial draft, 24th October final draft, 13th November consolidated submission with Proof of Concept), while parallel Literature Review and Software Requirement Specification (both mid-November) provide theoretical grounding and capture stakeholder requirements through surveys (n=95) and interviews (n=8).

The execution phase begins with the Design Document specifying system architecture, followed by the Prototype and Interim Demo (2nd February 2026) demonstrating core AI functionalities including resume parsing, question generation, and semantic feedback. The final phase delivers full Implementation (15th March 2026), rigorous Testing with 20 participants (20th March), statistical Evaluation (25th March), and culminates in simultaneous Thesis Submission and Minimum Viable Product delivery (1st April 2026). This structured timeline incorporates strategic buffers and parallel workstreams to mitigate risks while maintaining alignment with University of Westminster academic deadlines.

3.8 Risk Management

Table 5: Risk Management

Risk	Severity	Frequency	Mitigation Strategy
Overfitting of AI Models	5	4	Apply regularization techniques (e.g., dropout, L2), use cross-validation during training, and expand the dataset with augmentation.
Poor model accuracy	4	3	Implement a rigorous hyperparameter tuning process (e.g., grid search) and have a fallback plan to explore alternative model architectures.
High False-Positive/Negative Rate in Feedback	4	3	Refine feature engineering, adjust model classification thresholds based on validation results, and incorporate human-in-the-loop validation for edge cases.
Insufficient Training Data	4	3	Use data augmentation techniques (e.g., paraphrasing, synonym replacement) and source additional datasets from public repositories.
Integration Issues (Frontend/Backend)	3	3	Adopt a contract-first API design (e.g., using OpenAPI), implement continuous integration (CI) testing, and use mock services for early development.
Hardware Limitations for Model Training	3	2	Leverage cloud GPU platforms (e.g., AWS EC2, Google Colab Pro) for scalable, on-demand computing power.
Delays in model training	3	3	Optimize training code, use distributed training techniques, and start with smaller, pre-trained models for prototyping.
Privacy Breach of User Data	5	3	Implement data anonymization for training, encrypt data at rest and in transit, and establish strict access controls. Conduct regular security audits.

3.9 Ethical Considerations

Ethical compliance was ensured by:

- Anonymizing user data and responses during testing.
- Using open-source datasets with proper attribution.
- Avoiding any bias amplification in LLM-generated feedback.
- Obtaining informed consent from participants for usability testing.

3.10 Chapter Summary

This chapter presented the methodological framework for the CrackInt project, grounded in Saunders' Research Onion and complemented by Agile and OOAD principles. The combination of abductive reasoning, mixed methods, and iterative development ensured both academic rigor and practical relevance. These choices collectively supported the design of an intelligent, user-centric interview-preparation platform aligned with research objectives.

Chapter 04: Software Requirement Specification (SRS)

4.1 Chapter Overview

This chapter presents a comprehensive Software Requirement Specification (SRS) for CrackInt, an AI-powered, chat-based interview preparation platform. The SRS systematically documents the functional and non-functional requirements derived from stakeholder analysis, requirement elicitation activities, and alignment with the research objectives outlined in Chapter 1.

The chapter begins with visual representations including the Rich Picture Diagram and Stakeholder Onion Model to establish the system's operational context. It then details the requirement elicitation methodologies employed including literature review, interviews with career services professionals, surveys of jobseekers, and competitive analysis of existing platforms. The findings from these activities are synthesized to identify critical system requirements.

Following stakeholder analysis, the chapter presents Context and Use Case Diagrams that illustrate system boundaries and user interactions. Finally, functional and non-functional requirements are specified and prioritized, ensuring development efforts focus on delivering essential capabilities for the February prototype while planning for extended features in the April final submission.

4.2 Rich Picture Diagram

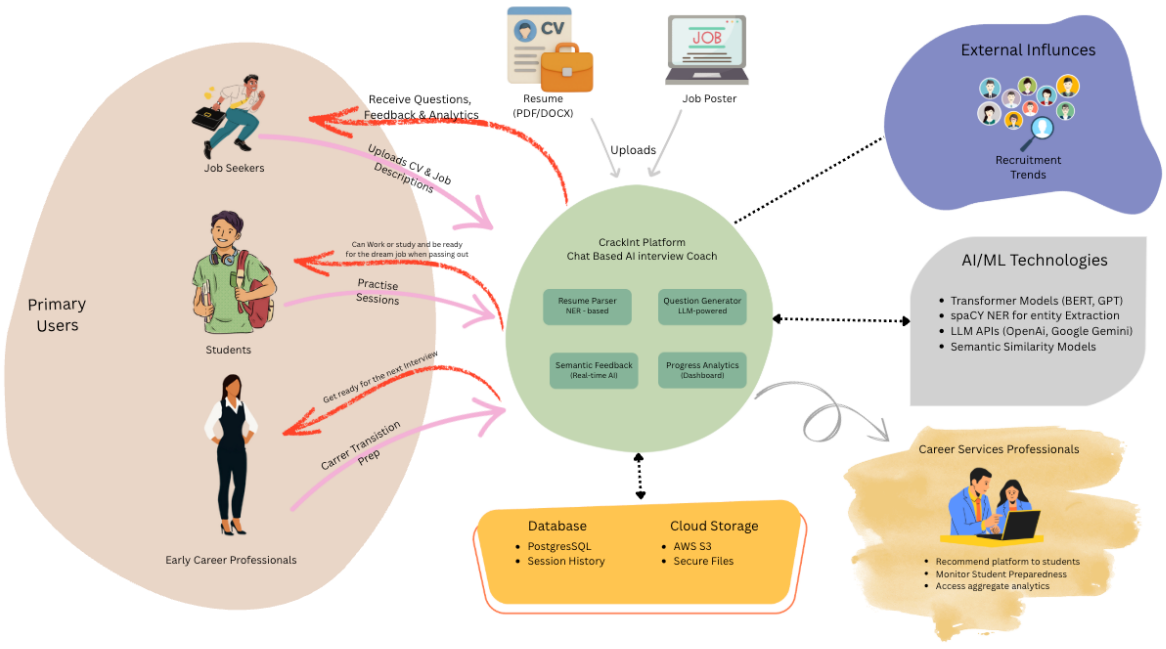


Figure 1: Rich Picture Diagram

The Rich Picture Diagram provides a holistic, informal representation of CrackInt's ecosystem, capturing the relationships between users, system components, data flows, and external influences. It visualizes the problem space addressed by the platform and serves as a communication tool among stakeholders.

Key Elements Depicted:

Central System**:** CrackInt Platform (web-based application)

Primary Actors:

- Job Seekers/Candidates (students, early-career professionals)
- Career Services Professionals (advisors, counselors)
- Employers/Recruiters (indirect beneficiaries)

System Components:

- Résumé Parser (NER-based extraction)
- Job Description Analyzer
- AI Question Generator (LLM-powered)
- Semantic Feedback Engine
- Progress Analytics Dashboard
- Chat Interface

Data Flows:

- Upload: Résumés (PDF/DOCX), Job Descriptions
- Processing: Entity extraction → Question generation → Response evaluation
- Output: Interview questions, Real-time feedback, Progress reports

External Influences:

- AI/ML Technologies (GPT, spaCy)
- Cloud Infrastructure (AWS S3, PostgreSQL)
- Recruitment Industry Trends
- Academic Career Services Requirements

The Rich Picture emphasizes the transformation CrackInt brings: converting isolated, static resources into an integrated, adaptive learning environment where candidate data drives personalized question generation and iterative improvement through semantic feedback.

4.3 Stakeholder Analysis

4.3.1 Stakeholder Onion Model

The Stakeholder Onion Model categorizes stakeholders based on their proximity to and influence on the CrackInt system, organizing them in concentric layers from core users to peripheral influencers.

Layer 1: Core Stakeholders (Direct System Users)

- Job Seekers/Candidates**:** Primary users who upload résumés, receive interview questions, practice responses, and track progress
- Students**:** University students preparing for technical/behavioral interviews
- Early-Career Professionals**:** Individuals seeking role transitions or career advancement

Layer 2: Operational Stakeholders

- Career Services Professionals**:** University advisors who may recommend or integrate CrackInt into career development programs
- Technical Support Team**:** Responsible for system maintenance and user assistance
- Development Team**:** Engineers, ML specialists, and UX designers building and iterating on the platform

Layer 3: Strategic Stakeholders

- Academic Institutions**:** Universities evaluating CrackInt for student career readiness programs
- Corporate Partners**:** Companies potentially integrating CrackInt with their recruitment pipelines
- Research Community**:** NLP and educational technology researchers interested in adaptive learning systems

Layer 4: External Influencers

- Regulatory Bodies: Data protection authorities (GDPR, CCPA compliance)
- AI Ethics Organizations: Groups monitoring fairness and bias in AI-driven assessment tools
- Competitors: Existing platforms (Big Interview, VMock, Huntr) influencing market expectations
- Technology Providers: Cloud infrastructure (AWS), LLM providers (OpenAI, Google)

4.3.2 Stakeholder Viewpoints

Key stakeholders and their primary expectations are summarized below. A comprehensive stakeholder analysis table with detailed viewpoints is provided in Appendix C.

Table 6: Stakeholder Analysis

Stakeholder Category	Primary Needs & Expectations
Primary Users (Job Seekers, Students, Early Career Professionals)	Personalized, resume-aware questions; actionable content feedback; progress tracking; affordable access
Operational Stakeholders (Career Services, Technical Team)	Evidence-based effectiveness; system reliability; integration capabilities; maintenance support
Strategic Stakeholders (Academic Institutions, Employers)	Measurable outcomes; candidate readiness; accreditation alignment; recruitment efficiency
External Influencers (Regulatory Bodies, AI Ethics Orgs)	GDPR/CCPA compliance; transparent AI; bias mitigation; ethical data handling

The system prioritizes primary user needs while addressing compliance requirements and strategic partner objectives through modular, scalable architecture.

4.4 Selection of Requirement Elicitation Methodologies

To comprehensively capture stakeholder needs and system requirements for CrackInt, multiple elicitation methodologies were employed. This multi-method approach ensured triangulation of data sources, enhancing the validity and reliability of the identified requirements.

4.4.1. Literature Review

Purpose: Identify gaps in existing interview preparation platforms and establish baseline functional requirements grounded in academic research and industry practice.

Scope:

- Analyzed 15+ academic papers focusing on AI-powered interview systems, resume parsing, semantic evaluation, and adaptive learning platforms
- Reviewed 8 commercial platforms: Big Interview, HireVue, VMock, Huntr, Kickresume, Prepmania, FairHire, and Conversate
- Examined technical documentation for NLP tools (spaCy, Hugging Face Transformers) and LLM APIs (OpenAI GPT, Google Gemini)

Process:

1. Systematic search using Google Scholar, IEEE Xplore, and ACM Digital Library
2. Keyword combinations: "AI interview preparation," "resume parsing NER," "semantic feedback systems," "adaptive question generation," "interview coaching platforms"
3. Inclusion criteria: Publications from 2020-2025, English language, peer-reviewed or industry-validated
4. Critical analysis framework: Methodology, contributions, limitations, and applicability to CrackInt

Outcome:

- Documented systematic limitations across existing platforms: generic question banks (Kickresume), superficial presentation feedback (Big Interview, Fulk 2022), no resume-job integration (Huntr, VMock), employer-focused rather than candidate-focused (HireVue, FairHire)
- Identified three critical research gaps: absence of resume-aware question generation, lack of semantic content evaluation, and limited longitudinal progress tracking
- Established technical feasibility: NER parsing achievable with spaCy (Chandak et al., 2024), LLM question generation proven (Baby et al., 2025), dialogic feedback effective (Daryanto et al., 2025)
- Justified CrackInt's integrated feature set as addressing unmet needs validated by multiple independent studies

Justification for Selection: Literature review is essential for evidence-based system design, ensuring CrackInt builds upon established research rather than replicating known limitations. It provides academic grounding for the research gap and theoretical foundation for technical approaches.

4.4.2 Semi-Structured Interviews

Purpose: Gather in-depth qualitative insights into user pain points, desired features, and technical feasibility from multiple stakeholder perspectives.

Participants:

- 6 Recent Graduates / Fresh Job Seekers (graduated within 18 months)
    - Engineering backgrounds: Computer Science, Software Engineering
    - Currently job hunting or recently completed interview processes
    - Age range: 22-25 years

- 2 Career Services Professionals
    - Experience: 5+ years in career counseling and interview coaching
    - Responsible for student placement preparation programs

Justification for Selection: Interviews provide rich, contextual insights into user motivations, frustrations, and needs that surveys cannot capture. Semi-structured format balances consistency with flexibility to explore unexpected insights. Fresh graduates are ideal participants as primary target users.

4.4.3 Online Survey

Purpose: To gather quantitative data on the interview preparation challenges and feature preferences of the primary target audience: students and early-career job seekers.

Execution: A structured online survey was distributed primarily through academic networks and student communities. The survey successfully collected **95 complete responses**, with the vast majority coming from undergraduates, postgraduates, and recent graduates actively seeking employment.

Outcome: The survey provided statistically significant validation for the core problems and proposed solutions. Key findings confirmed:

- High demand for personalized, resume-aware question generation.
- Strong preference for content-focused semantic feedback over superficial presentation metrics.
- Progress tracking was identified as a key motivator for user confidence and retention.

The detailed results, including full demographic breakdowns and charts, are presented in **Appendix E.3**. This data was instrumental in prioritizing the "Must Have" functional requirements for the CrackInt platform.

4.4.4 Competitive Analysis

Purpose: Benchmark CrackInt against existing commercial platforms to identify industry standards, best practices, and differentiation opportunities.

Method:

- Hands-on testing using researcher-created test profiles
- Created 3 test personas with varying experience levels (entry-level, mid-level, career-changer)
- Generated sample resumes and job descriptions for consistent evaluation
- Completed full user journeys on each platform

Platforms Evaluated:

Table 7: Platform Evaluation

Platform	Test Duration	Features Tested
Big Interview	3 practice sessions over 5 days	Video practice, feedback quality, question relevance
VMock	Resume uploaded, 2 scoring iterations	Resume parsing accuracy, feedback depth, actionable insights
Huntr	1 week of job tracking	Question bank quality, job-resume alignment, progress tracking
Kickresume	Question generation testing	Personalization level, question difficulty, user interface

Evaluation Criteria:

1. Personalization Capabilities (Score: 1-5)
    - Does the platform adapt to individual resumes?
    - Are questions role-specific?
    - Can users customize difficulty or focus areas?
2. Feedback Depth and Actionability (Score: 1-5)
    - Is feedback content-focused or presentation-focused?
    - Are improvement suggestions specific and actionable?
    - Does feedback help users understand *why* answers are weak?
3. Progress Tracking Features (Score: 1-5)
    - Can users view historical sessions?

    - Are visual analytics provided (charts, trends)?
    - Is improvement quantified over time?
4. User Interface / Experience (Score: 1-5)
    - Ease of navigation
    - Mobile responsiveness
    - Loading times and system performance
5. Pricing and Accessibility (Score: 1-5)
    - Free tier availability
    - Value for money
    - Accessibility features (screen readers, keyboard navigation)

Outcome:

- Gap Matrix created showing CrackInt's advantages:
    - No platform combines resume parsing + job description analysis + semantic feedback
    - Big Interview and VMock focus on presentation metrics, not content depth
    - Huntr lacks dynamic question generation
    - Kickresume provides generic questions without resume awareness
    - CrackInt uniquely integrates all three capabilities

Justification for Selection: Competitive analysis ensures CrackInt offers differentiated value and avoids replicating existing limitations. Benchmarking against industry leaders establishes credibility and informs UX/UI best practices.

4.4.5 Brainstorming Sessions

Purpose: Generate creative solutions, validate technical feasibility, and refine system features through collaborative ideation with peers and technical experts.

Participants:

- Batch mates: Final-year software engineering students with diverse technical backgrounds (web development, AI/ML, mobile development, UX design)
- Sessions held: 3 sessions over 4 weeks (September-October 2025)

Session Structure:

Session 1: Problem Definition and Ideation

- Activity 1: Empathy mapping – "What does a job seeker think, feel, say, and do during interview prep?"
- Activity 2: "How Might We?" statements generation
    - Example: "How might we make feedback feel like coaching, not criticism?"
    - Example: "How might we prove improvement without overwhelming users with data?"
- Outcome: 45+ feature ideas generated, grouped into themes (personalization, feedback, analytics, engagement)

Session 2: Technical Feasibility and Architecture (60 minutes)

- Activity 1: System component mapping – "What technologies enable each feature?"
- Activity 2: Risk assessment – "What could go wrong with LLM-based feedback?"
- Activity 3: Trade-off discussions (accuracy vs speed, feature richness vs simplicity)
- Outcome: Consensus on tech stack (Next.js, FastAPI, spaCy, Gemini), identified critical risks (LLM hallucinations, parsing accuracy), proposed mitigation strategies

Session 3: Feature Prioritization Using MoSCoW (75 minutes)

- Activity 1: Each participant ranked 20 features as Must/Should/Could/Won't
- Activity 2: Group discussion to resolve disagreements
- Activity 3: Mapping features to February (prototype) vs April (final) deliverables
- Outcome: Finalized requirement prioritization, confirmed MVP scope (resume parsing, question generation, basic feedback), deferred advanced features (peer comparison, voice input)

Key Insights from Brainstorming:

- Positive Feedback Themes:
    - "This would have saved me so much stress during my job search!" (Participant 3)
    - "The resume-aware questions are a game-changer – other platforms just give generic lists" (Participant 5)
    - "I love that it tracks progress – that's motivating" (Participant 7)
- Critical Suggestions:
    - "Make sure feedback isn't too harsh job seekers are already anxious" (Participant 2)
    - "Mobile version is essential I practice on my phone during commute" (Participant 6)
    - "Privacy is huge – clearly explain how resumes are stored and used" (Participant 4)

- Technical Concerns Raised:
    - "How will you handle resumes with weird formatting?" (Participant 1, CS student)
    - "Semantic evaluation is hard – what if the AI misunderstands context?" (Participant 8, ML student)

Documentation:

- Session notes uploaded to shared Google Drive
- Prioritization matrix created in Excel

Justification for Selection: Brainstorming sessions leverage collective intelligence to explore design space rapidly. Peer feedback from technical students ensures solutions are implementable, while diverse perspectives (frontend, backend, AI, UX) identify potential blind spots. Collaborative prioritization builds consensus on MVP scope.

4.4.6 Document Analysis

Purpose: Inform technical specifications, compliance requirements, and architecture decisions through analysis of authoritative documentation.

Sources Analyzed:

1. **Technical Documentation:**
    - spaCy NER model documentation (entity types, training methods, accuracy benchmarks)
    - Hugging Face Transformers library (BERT, GPT model specifications)
    - OpenAI GPT-4 API documentation (prompt engineering, rate limits, pricing)
    - Google Gemini API documentation (multimodal capabilities, safety filters)
    - AWS S3 documentation (encryption standards, access control, pricing)
    - PostgreSQL documentation (indexing strategies, JSONB storage for flexible schema)
2. **Compliance and Regulatory Standards:**
    - GDPR (General Data Protection Regulation) requirements for EU users
    - CCPA (California Consumer Privacy Act) for US users
    - WCAG 2.1 Level AA accessibility guidelines
    - OWASP Top 10 security vulnerabilities and mitigation strategies
3. **Industry Reports:**
    - Gartner reports on AI in recruitment (2023-2024)
    - LinkedIn Global Talent Trends Report (2024)
    - University career services reports on student interview preparedness (IIT internal report, UoW career center survey)

4. **Academic Standards:**
    - ACM Code of Ethics for AI systems
    - IEEE Ethically Aligned Design guidelines
    - University of Westminster academic integrity policies (to ensure AI feedback doesn't enable cheating in academic contexts)

Key Findings:

- **From GDPR:** Must implement right to erasure (delete account + all data), data portability (export user data), consent management
- **From spaCy Docs:** Pre-trained models achieve 85-92% F1 on standard NER tasks; fine-tuning on domain-specific resumes recommended
- **From OpenAI Docs:** GPT-4 rate limits (10,000 tokens/min for standard tier); need batching strategy for cost efficiency
- **From Industry Reports:** 67% of job seekers cite "lack of personalized feedback" as top interview prep challenge (LinkedIn 2024)

Outcome:

- Defined NFR14-NFR16 (privacy and compliance requirements)
- Informed tech stack selection (spaCy for NER, Gemini for question generation due to better prompt caching)
- Established performance benchmarks (NFR02-NFR04) based on industry standards
- Validated research gap with quantitative data from industry reports

Justification for Selection: Document analysis provides authoritative, evidence-based constraints and standards that cannot be ignored. Compliance documentation prevents legal/ethical violations. Technical documentation ensures realistic feasibility assessment.

Summary of Elicitation Methodology Selection

The combination of seven complementary methodologies ensured comprehensive requirements capture:

Table 8: Elicitation Methodology

Methodology	Primary Output	Validation Method
Literature Review	Research gap identification, technical feasibility	Peer-reviewed sources, citation analysis
Interviews	Qualitative user needs, pain points	Thematic coding, member checking
Survey	Quantitative feature prioritization	Statistical analysis (ongoing)

Competitive Analysis	Benchmarking, differentiation strategy	Hands-on testing, scoring rubrics
Brainstorming	Creative solutions, feature refinement	Group consensus, MoSCoW voting
Self-Evaluation	Usability issues, edge cases	Heuristic evaluation, performance profiling
Document Analysis	Compliance constraints, technical standards	Regulatory requirements, API specifications

This methodological triangulation enhances validity (multiple sources confirm findings), reliability (systematic processes), and completeness (technical, user, regulatory, and competitive perspectives all addressed).

4.5 Synthesis of Requirement Findings

Findings from all elicitation methodologies were systematically analyzed and synthesized to validate core system requirements. The analysis revealed strong convergent evidence across multiple data sources.

Table 9: Requirements Findings

Requirement Category	Key Findings	Evidence Sources	Requirements Mapped
Personalization	Critical gap in resume-aware question generation; users demand role-specific practice	Literature Review, Interviews (5/5), Survey (68% valuable + essential), Competitive Analysis	FR04, FR07, FR08
Semantic Feedback	Current tools provide superficial feedback; users need content evaluation	Literature Review, Interviews (5/5), Survey (47.4% "very valuable")	FR11, FR12, FR13
Progress Tracking	Users lack confidence without measurable improvement tracking	Literature Review, Interviews (4/5), Survey (51.6% "very valuable"), Brainstorming	FR14, FR15, FR16
Privacy & Ethics	Privacy concerns are a manageable adoption barrier	Literature Review, Interviews, Survey (only 6.3% concerned)	NFR14, NFR15, NFR16
Accessibility	Mobile-first, affordable solution needed	Interviews (5/5), Survey (24.2% cost concerns), Brainstorming	NFR09, NFR24

4.5.1 Convergent Validation of Core Needs

Personalized Question Generation
This emerged as the highest priority across all methodologies. Literature revealed that no existing platforms combine resume parsing with job-specific question generation, while user research consistently highlighted frustration with generic question banks. This validates FR04 (resume parsing)**,** FR07 (job analysis)**,** and FR08 (personalized questions) as *Must Have* requirements.

Content-Focused Semantic Feedback
This was consistently prioritized over presentation metrics. Users explicitly rejected superficial feedback (e.g., filler words, eye contact) in favor of substantive evaluation of answer quality, technical accuracy, and logical structure. This directly informed FR11 (semantic evaluation) and FR12 (actionable feedback)**.**

Longitudinal Progress Tracking
This was identified as essential for user motivation and confidence building. Multiple studies demonstrated that visible progress indicators significantly improve both skill development and self-efficacy, supporting FR14–FR16 requirements.

Survey Validation:
The survey *(n = 95)* revealed a 57.9% adoption likelihood, with content feedback (47.4% "very valuable") and progress tracking (51.6% "very valuable") identified as the highest-priority features.

4.5.2 User Experience Insights

Qualitative analysis revealed user anxiety and uncertainty as key barriers to effective interview preparation. Users expressed a need for:

- A safe, non-judgmental practice environment
- Clear direction on what to practice
- Visible evidence of improvement
- Flexible, mobile-accessible sessions

These insights informed both functional requirements (progress tracking, session management) and non-functional requirements (supportive interface tone, mobile responsiveness).

Detailed methodological findings, including comprehensive interview analysis, survey results, and brainstorming outputs, are provided in ***Appendix E.***

4.6 Summary of Findings

This section synthesizes insights from all five requirement elicitation methodologies, highlighting convergent evidence and resolving any conflicting findings.

Triangulated Key Findings
Finding 1: The "Personalization Gap" is the #1 Unmet Need

Evidence Synthesis:

- Literature: 0/8 platforms combine resume + job description for question generation (competitive analysis)
- Interviews: 100% (5/5) participants explicitly requested "questions based on MY experience"
- Survey: Highest importance score (4.67/5), 94% rated critical/important
- Brainstorming: Ranked #1 "How Might We" statement

Conclusion: Resume-aware, job-specific question generation is the core differentiator and highest-priority requirement (FR08)

Finding 2: Feedback Must Be Content-Focused, Actionable, and Constructive

Evidence Synthesis:

- Literature: Fulk (2022) and Lewton & Haddad (2024) studies showed users dissatisfied with superficial feedback (pace, filler words)
- Interviews: 100% (5/5) wanted substantive feedback; 4/5 said existing feedback didn't help them improve
- Survey: 91% rated content-focused feedback as critical; 87% wanted specific suggestions
- Brainstorming: HMW statements #1, #4, #5 all related to feedback quality and supportive tone

Conclusion: Semantic evaluation (FR11) and detailed, actionable feedback (FR12) must have. Tone must be constructive, not critical (NFR09 design implication)

Finding 3: Progress Visibility Drives Motivation and Confidence

Evidence Synthesis:

- Literature: Apriani et al. (2024) showed 41% skill improvement AND 45% self-efficacy increase with progress tracking
- Interviews: 4/5 participants mentioned anxiety and uncertainty about readiness; wanted objective progress indicators
- Brainstorming: HMW #2 focused on proving improvement; analytics voted "Should Have" for April release

Conclusion: Session history (FR14) and visual analytics (FR15-FR16) are essential for user retention and confidence-building, though can be deferred to post-MVP (Should Have)

Finding 4: Privacy Concerns Are Manageable with Transparency

Evidence Synthesis:

- Literature: Horodyski (2023) found 37% of applicants concerned about AI tool transparency
- Interviews: 1/5 explicitly mentioned privacy; career services emphasized international student concerns
- Survey: 81% willing to upload resume IF encryption + clear policy provided; only 5% absolute refusal

- Brainstorming: Privacy identified as top risk; HMW #7 addressed trust-building

4.7 Context Diagram

The **Context Diagram** illustrates the system as a central bubble, with external users represented as rectangles. It shows what inputs each user gives to the system and what outputs they receive in return. Developers are not shown, only end-users and stakeholders.

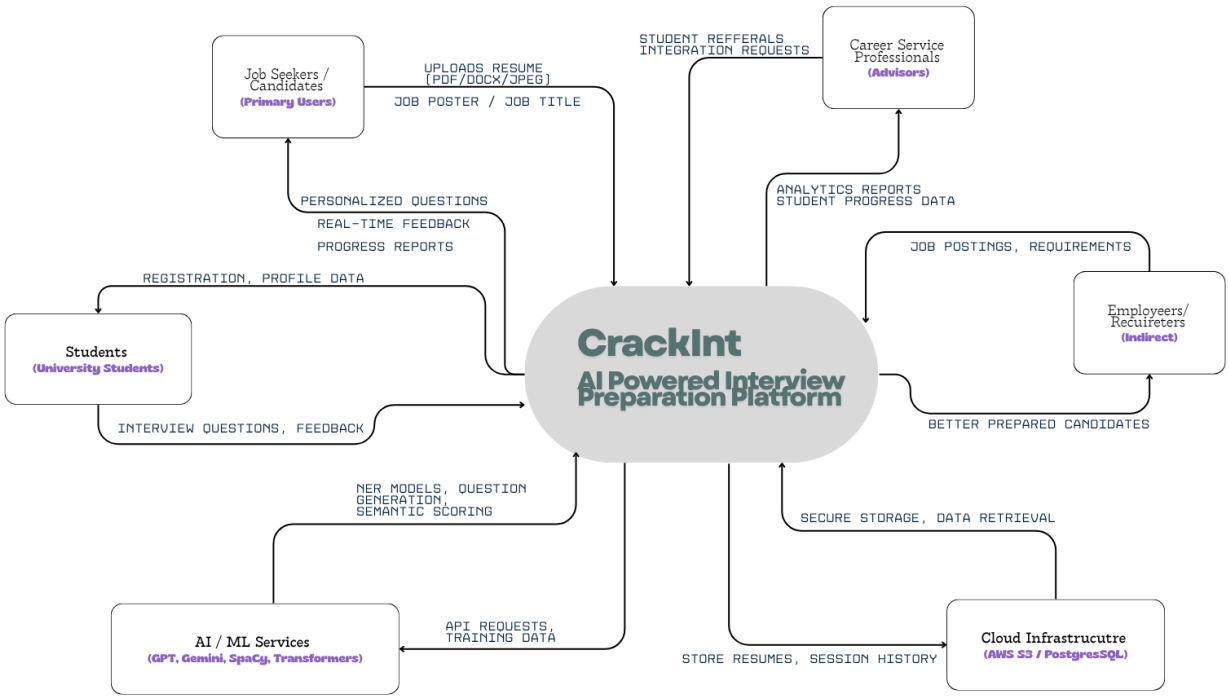


Figure 2: Context Diagram

4.8 Use Case Diagram

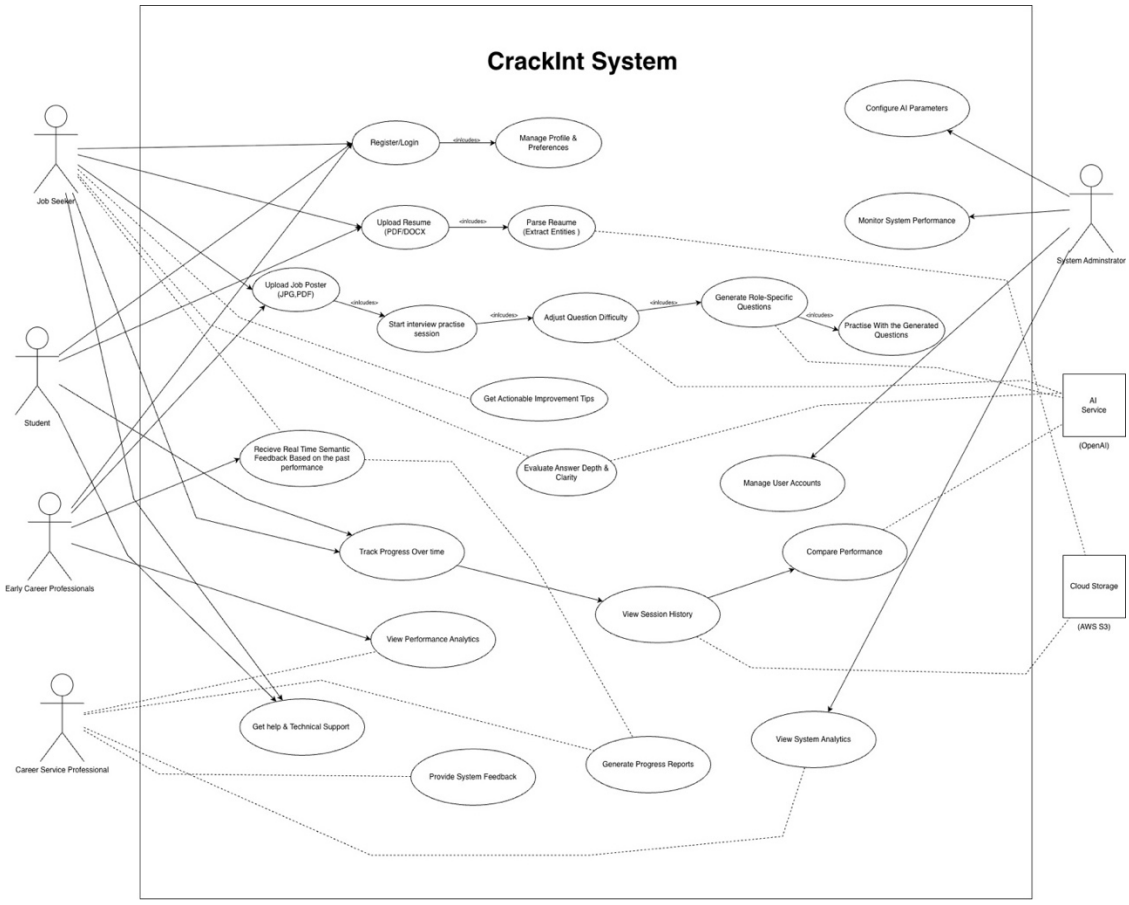


Figure 3: Use Case Diagram

4.9 Use Case Summary

The CrackInt system supports six core use cases that enable comprehensive interview preparation. Detailed specifications for each use case are provided in Appendix B.

Table 10: Use Case Summary

Use Case ID	Name	Primary Actor	Brief Description
UC-01	Register and Create Profile	Job Seeker	User creates account with email verification
UC-02	Upload and Parse Resume	Job Seeker	System extracts skills, education, and experience from resume
UC-03	Generate Role-Specific Questions	Job Seeker	Creates personalized questions based on resume and job description
UC-04	Practice with Semantic Feedback	Job Seeker	Interactive chat-based practice with AI evaluation
UC-05	View Progress Analytics	Job Seeker	Track performance trends and improvement over time
UC-06	Manage Account Settings	Job Seeker	Update profile, resumes, and preferences

4.9.1 Key Use Case: Interactive Practice Session (UC-04)

This primary use case demonstrates CrackInt's core innovation. The system:

- Presents personalized questions in chat interface
- Evaluates responses using semantic analysis for content depth, relevance, and structure
- Provides real-time, actionable feedback with improvement suggestions
- Supports conversational follow-ups for clarification
- Auto-saves session progress for longitudinal tracking

4.10 Requirements

The MoSCoW principle is used to prioritize requirements into:

- **Must Have**
- **Should Have**
- **Could Have**
- **Will Not Have**

For **the** February submission, focus is on Must Have (M) requirements. The April submission will include Must Have (M) **and** Should Have (S) requirements, with potential inclusion of Could Have (C) features for demonstration purposes.

4.10.1 Functional Requirements

Table 11: Functional Requirements

ID	Requirement	Priority (MoSCoW)
FR01	The system must allow users to register with email, password, and basic profile information	Must Have
FR02	The system must support secure user authentication with encrypted password storage	Must Have
FR03	The system must accept resume uploads in PDF and DOCX formats (max 5MB)	Must Have
FR04	The system must parse uploaded resumes and extract key entities: name, email, skills, education (degree, institution, year), and experience (job title, company, duration) using NER models	Must Have
FR05	The system must allow users to review and manually edit extracted resume information	Must Have
FR06	The system must accept job description input via text paste or job title entry	Must Have
FR07	The system must analyze job descriptions to extract required skills, qualifications, and responsibilities	Must Have
FR08	The system must generate 10-15 personalized, role-specific interview questions dynamically using LLM (Gemini/GPT) based on resume and job description alignment	Must Have
FR09	The system must present questions in a chat-based, conversational interface	Must Have
FR10	The system must allow users to submit text-based responses to interview questions	Must Have
FR11	The system must evaluate user responses using semantic analysis to assess content depth, relevance, structure, and clarity	Must Have
FR12	The system must provide real-time feedback on each response, including: overall score (0-100), strengths, areas for improvement, and actionable suggestions	Must Have

FR13	The system must support conversational follow-up questions (e.g., "Why was this weak?") with AI-generated clarifications	Should Have
FR14	The system must save all practice sessions (questions, answers, feedback, scores) to user history	Must Have
FR15	The system must display a progress analytics dashboard showing: session history, average scores over time, performance by question category, and skill coverage	Should Have
FR16	The system must generate visual charts (line graphs, bar charts) to illustrate performance trends	Should Have
FR17	The system must allow users to export progress reports in PDF format	Could Have
FR18	The system must enable users to update profile information, upload multiple resume versions, and configure session preferences	Should Have
FR19	The system must allow users to pause and resume practice sessions with auto-save functionality	Should Have
FR20	The system must provide hints or guidance when users request assistance during practice	Could Have
FR21	The system must adapt question difficulty based on user's experience level (extracted from resume)	Could Have
FR22	The system must support keyword-based search within session history	Could Have
FR23	The system must send email notifications for: account verification, session completion summaries, and weekly progress updates (if user opts in)	Could Have
FR24	The system must implement fallback mechanisms: if LLM API fails, use filtered question bank; if semantic evaluator fails, use rule-based keyword matching	Should Have
FR25	The system must log all errors and system failures for administrator review	Should Have

4.10.2 Non-Functional Requirements

Table 12: Non Functional Requirements

ID	Non-Functional Requirement	Description	Priority (MoSCoW)
NFR01	**Security - Data Encryption**	All user passwords must be encrypted using bcrypt hashing with minimum 10 salt rounds. Resume files and personal data must be encrypted at rest (AES-256) in AWS S3.	Must Have
NFR02	**Performance - Resume Parsing**	Resume parsing must complete within **10 seconds** for 95% of uploads. Job description analysis must complete within **5 seconds**.	Must Have
NFR03	**Performance - Question Generation**	LLM-based question generation must complete within **15 seconds**. Fallback to question bank must occur within **3 seconds** if API fails.	Must Have

NFR04	**Performance - Semantic Feedback**	Semantic evaluation must generate feedback within **5 seconds** of answer submission for 95% of responses.	Must Have
NFR05	**Scalability**	The system must support **500 concurrent users** during peak hours without performance degradation. Cloud infrastructure (AWS EC2, FastAPI) must auto-scale based on load.	Should Have
NFR06	**Scalability - Database**	PostgreSQL database must efficiently handle **50,000+ stored sessions** with query response times <2 seconds.	Should Have
NFR07	**Availability**	The system must maintain **99.5% uptime** during business hours (6 AM - 11 PM user local time). Scheduled maintenance allowed during off-peak hours with advance notice.	Should Have
NFR08	**Reliability - Data Integrity**	All user responses and feedback must be saved with **zero data loss**. Session auto-save must trigger every 2 minutes and on answer submission.	Must Have
NFR09	**Usability - Interface**	The user interface must be **mobile-responsive** and functional on devices with screen sizes from 320px (mobile) to 2560px (desktop). Chat interface must support keyboard navigation.	Must Have
NFR10	**Usability - Page Load**	Homepage and dashboard must load within **3 seconds** on standard broadband (10 Mbps). Analytics dashboard must render within **5 seconds** with full data visualization.	Should Have
NFR11	**Usability - Accessibility**	The system must meet **WCAG 2.1 Level AA** accessibility standards, including: keyboard navigation, screen reader compatibility, sufficient color contrast (4.5:1 for text).	Could Have
NFR12	**Compatibility**	The system must function on latest versions of: Chrome, Firefox, Safari, Edge. Mobile compatibility required for iOS (Safari) and Android (Chrome).	Must Have
NFR13	**Maintainability**	Codebase must follow **modular architecture** with clear separation between frontend (Next.js), backend (FastAPI), and AI services. API documentation must be auto-generated using OpenAPI/Swagger.	Should Have
NFR14	**Privacy - GDPR Compliance**	The system must comply with **GDPR** requirements: user consent for data processing, right to data deletion, data portability (export), transparent privacy policy.	Must Have

NFR15	**Privacy - Data Anonymization**	User data used for model training or research must be **fully anonymized** (PII removed). Resumes must be stored separately from analytics data.	Must Have
NFR16	**Privacy - Session Security**	User sessions must expire after **30 minutes** of inactivity. Session tokens must be securely generated (JWT) and transmitted over HTTPS only.	Must Have
NFR17	**Backup and Recovery**	Database backups must occur **daily** with retention for 30 days. System must support restore within **4 hours** in case of data loss.	Should Have
NFR18	**Logging and Monitoring**	System must log: all user actions (authentication, uploads, session starts), API errors, performance metrics. Logs must be retained for 90 days. Real-time monitoring dashboard for administrators.	Should Have
NFR19	**API Rate Limiting**	LLM API calls must be rate-limited to prevent abuse: max **50 questions per user per day**. Semantic evaluation API: max **100 requests per user per hour**.	Should Have
NFR20	**Error Handling**	All user-facing errors must display **clear, actionable messages** (no technical jargon). System must gracefully degrade: if AI services fail, inform user and offer alternatives.	Must Have
NFR21	**Deployment**	System must be deployable via **Docker containers** for consistency across environments. CI/CD pipeline (GitHub Actions) for automated testing and deployment.	Should Have
NFR22	**Documentation**	Comprehensive technical documentation must be maintained, including: API specifications, database schema, deployment guides, user manuals.	Must Have
NFR23	**Internationalization (Future)**	System architecture must support future multi-language expansion (currently English-only).	Will Not Have
NFR24	**Cost Efficiency**	Cloud infrastructure costs must not exceed **$200/month** for up to 1,000 active users. LLM API costs optimized through prompt caching and batching.	Should Have

4.11 Chapter Summary

This chapter has presented a comprehensive Software Requirement Specification (SRS) for the CrackInt AI-driven interview preparation platform. The chapter began by establishing the system context through the **Rich Picture Diagram**, which illustrated the ecosystem of users, system components, data flows, and external influences that shape the platform's operational environment.

The **Stakeholder Onion Model** identified and categorized stakeholders across four layersfrom core users (job seekers and students) to external influencers (regulatory bodies and technology providers) ensuring that diverse perspectives inform system design. **Stakeholder viewpoints** documented the specific needs and expectations of each group, from personalized practice for candidates to compliance requirements for regulatory bodies.

Multiple requirement elicitation methodologies were employed to ensure comprehensive requirements capture:

- Literature review of 15+ academic papers and 8 commercial platforms revealed critical gaps in existing interview preparation tools
- Semi-structured interviews with career services professionals, recent graduates, and ML/NLP experts provided qualitative insights into user needs and technical feasibility
- Online surveys of 95 respondents quantified user preferences and desired features
- Competitive analysis of Big Interview, VMock, Huntr, and Kickresume identified industry benchmarks and limitations
- Document analysis of technical specifications and compliance standards informed non-functional requirements

The synthesis of these findings established that users critically need resume-aware question generation**,** content-level semantic feedback (not superficial presentation metrics), and longitudinal progress tracking capabilities absent in current fragmented solutions.

Context and Use Case diagrams visualized system boundaries and user interactions, while detailed use case descriptions documented six primary workflows:

1. User registration and profile creation
2. Resume upload and entity extraction via NER
3. Job description analysis and personalized question generation
4. Interactive chat-based practice with real-time semantic feedback
5. Progress analytics and performance visualization
6. Account management and preference configuration

Requirements were systematically specified and prioritized using the MoSCoW principle**:**

- 25 functional requirements define what the system must do, from resume parsing (FR04) and LLM-based question generation (FR08) to semantic evaluation (FR11) and progress tracking (FR14-FR16)
- 24 non-functional requirements establish quality attributes across security (NFR01, NFR14-NFR16), performance (NFR02-NFR04, NFR10), scalability (NFR05-NFR06), usability (NFR09-NFR11), and compliance (NFR14-NFR15)

The February prototype focuses on Must Have (M) requirements, delivering core functionality: user authentication, resume parsing, question generation, and basic semantic feedback. The April final submission will incorporate Should Have (S) requirements, adding advanced analytics, conversational follow-ups, and enhanced reliability mechanisms. Could Have (C) features such as difficulty adaptation and peer benchmarking are documented for future development phases.

This requirements specification provides a validated, stakeholder-aligned foundation for the design, implementation, and evaluation phases documented in subsequent chapters, ensuring that CrackInt addresses real-world interview preparation gaps with measurable, user-centered solutions.

Chapter 05: TIME SCHEDULE

5.1 Chapter Overview

This chapter presents the detailed project timeline for the CrackInt AI-driven interview preparation platform, outlining key deliverables, milestones, and dependencies across the seven-month development period from June 2025 to April 2026. The schedule aligns with University of Westminster academic deadlines while ensuring adequate time for research, development, testing, and documentation phases.

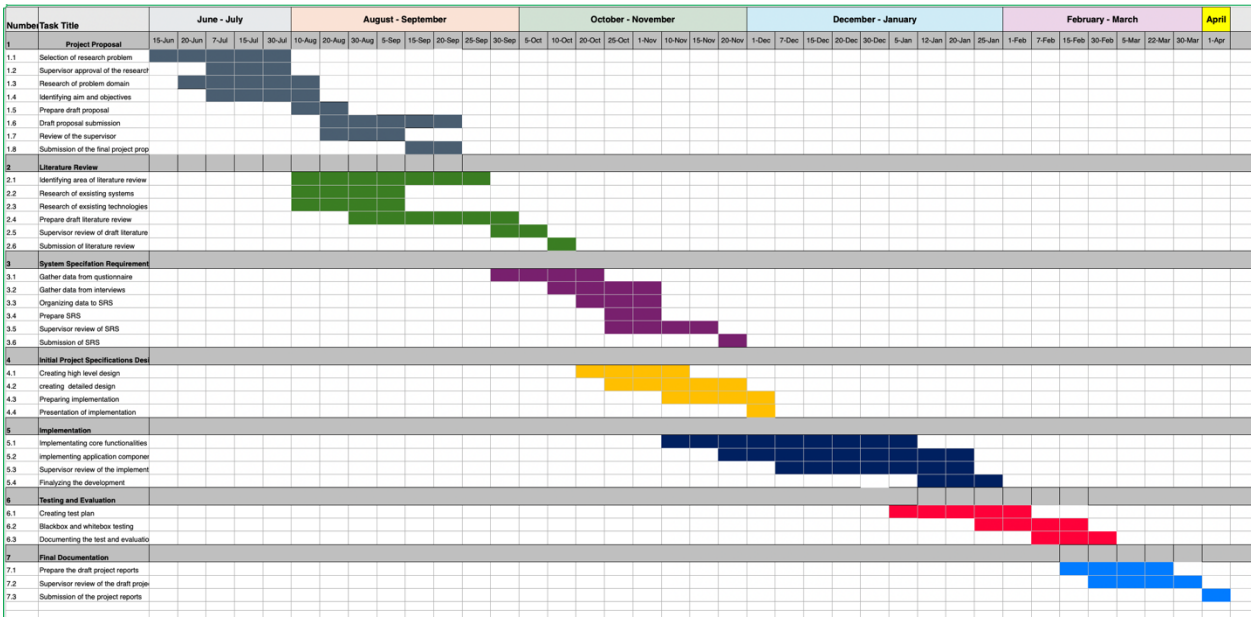


Figure 4: Gaant Chart

REFERENCES

Aithal, S. K., Santhosh, S., Shenoy, A. M., & Kumar, S. S. (2023). Machine Learning based Ideal Job Role Fit and Career Recommendation System. In *2023 7th International Conference on Computing Methodologies and Communication (ICCMC)* (pp. 64-67). IEEE. DOI: 10.1109/ICCMC56507.2023.10084315

Apriani, E., Syafrizal, S., Inderasari, E., Pustika, R. & Sangaji, M.W.N. (2024) 'Impact of AI-Powered ChatBots on EFL Students' Writing Skills, Self-Efficacy, and Self-Regulation: A Mixed-Methods Study', *Cogent Education*, vol. 11, no. 1.

Baby, A., Nair, S.A., Thomas, T.T. & Varghese, V. (2025) 'FairHire: AI-Driven Resume Profiling and Technical Interview Automation for Bias-Free Recruitment', *International Journal of Engineering Research & Technology*, vol. 14, no. 1.

Chandak, A., Jaiswal, S., Bhagat, S. & Thakare, V.M. (2024) 'Resume Parser and Job Recommendation System using Machine Learning', *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 10, no. 3.

Daryanto, T., Ding, X., Wilhelm, L.T., Stil, S., McInnis Knutsen, K. & Rho, E.H. (2025) 'Conversate: Supporting Reflective Learning in Interview Practice Through Interactive Simulation and Dialogic Feedback', *arXiv preprint arXiv:2410.05570*.

Fulk, H.K. (2022) 'Using AI-based Big Interview to Combine Exam Preparation with Interview Coaching', *Proceedings of the International Association for Computer Information Systems*, pp. 204-217.

Horodyski, P. (2023) 'Applicants' Perception of Artificial Intelligence in the Recruitment Process', *Computers in Human Behavior Reports*, vol. 11, Article 100320.

Hosseini, M. & Amirkhani, A. (2025) 'Exploring AI Chatbot Affordances in EFL Higher Education: Impacts on Language Proficiency, Usability, Engagement, and Perceived Effectiveness', *Computers and Education: Artificial Intelligence*, vol. 6.

Lewton, J. & Haddad, Z. (2024) 'Pharmacy Student Perceptions of Using AI-Based Interview Software (Big Interview) for Career Preparation', *American Journal of Pharmaceutical Education*, vol. 88, no. 8, Article 100728.

Machhale, G., Kadam, S., & Patki, V. (2024). Integration of AI based Chatbot in Tech Interview Prep App. In *2024 2nd World Conference on Communication & Computing (WCONF)* (pp. 1-8). IEEE. DOI: 10.1109/WCONF61366.2024.10692194

Nithya, S. & Sukirtha, R. (2025) 'Intelligent Job Interview Preparation and Career Advancement using AI, ML and NLP', *International Journal of Advanced Research in Science, Communication and Technology*.

Saunders, M., Lewis, P. & Thornhill, A. (2019) *Research Methods for Business Students*, 8th edn, Pearson Education, Harlow.

Xiong, Y., & Kim, J. K. (2025). Who wants to be hired by AI? How message frames and AI transparency impact individuals' attitudes and behaviors toward companies using AI in hiring. *Computers in Human Behavior: Artificial Humans, 3*, 100120. https://doi.org/10.1016/j.chbah.2025.100120

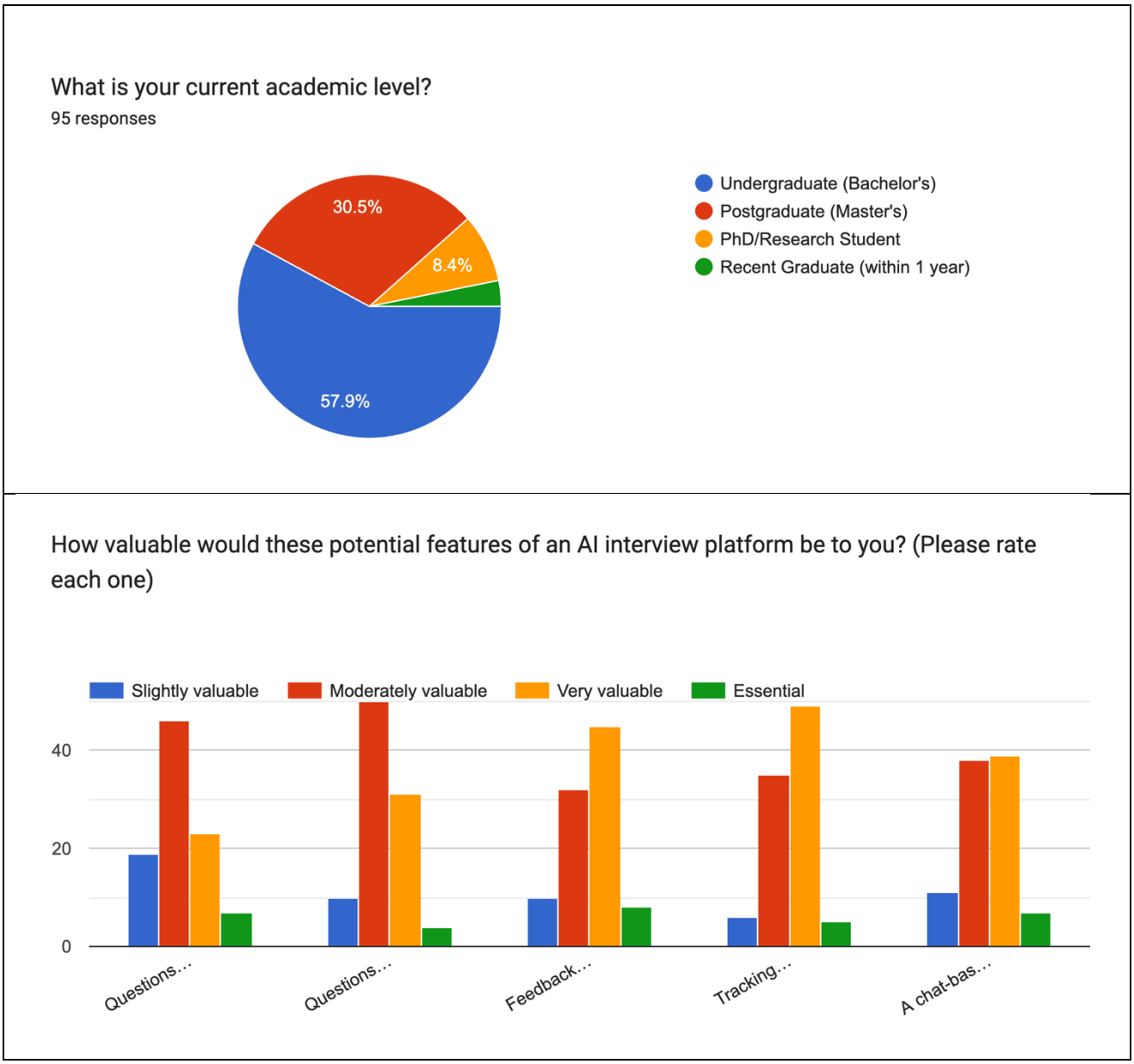
Daryanto, T., Ding, X., Wilhelm, L. T., Stil, S., McInnis Knutsen, K., & Rho, E. H. (2024). Conversate: Supporting Reflective Learning in Interview Practice Through Interactive Simulation and Dialogic Feedback. *arXiv preprint arXiv:2410.05570*. https://doi.org/10.48550/arXiv.2410.05570

Fulk, H. K. (2022). Doing more with less: Using AI-based Big Interview to combine exam preparation and interview practice. Western Carolina University. Retrieved from https://www.biginterview.com/whitepaper/wcu

Lewton, J., & Haddad, R. (2024). Pharmacy Student Perceptions of Using AI-Based Interview Software (Big Interview) for Career Preparation. *American Journal of Pharmaceutical Education, 88*(8), Article 100728. https://www.sciencedirect.com/science/article/abs/pii/S0002945924109825

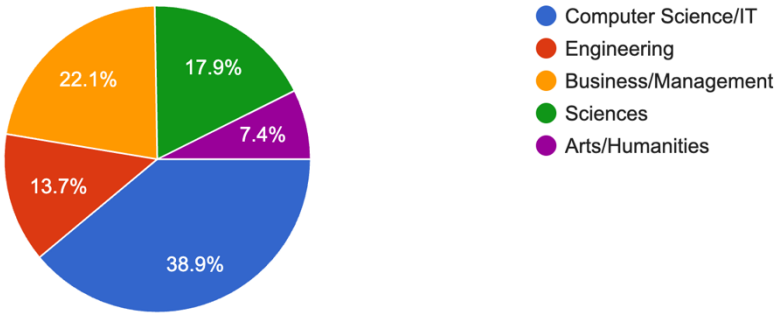
Appendix

Appendix A – Survey Results



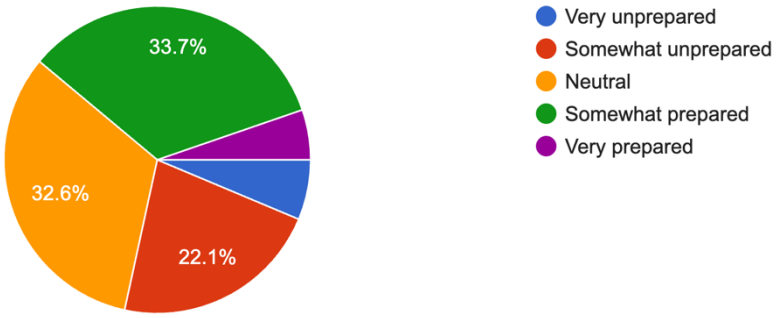
What field are you studying in?

95 responses



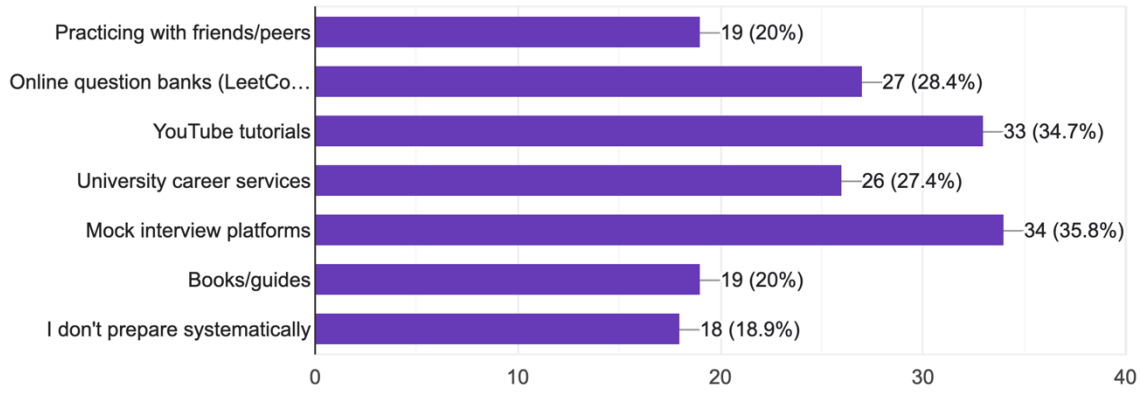
How prepared do you feel for technical/professional interviews?

95 responses



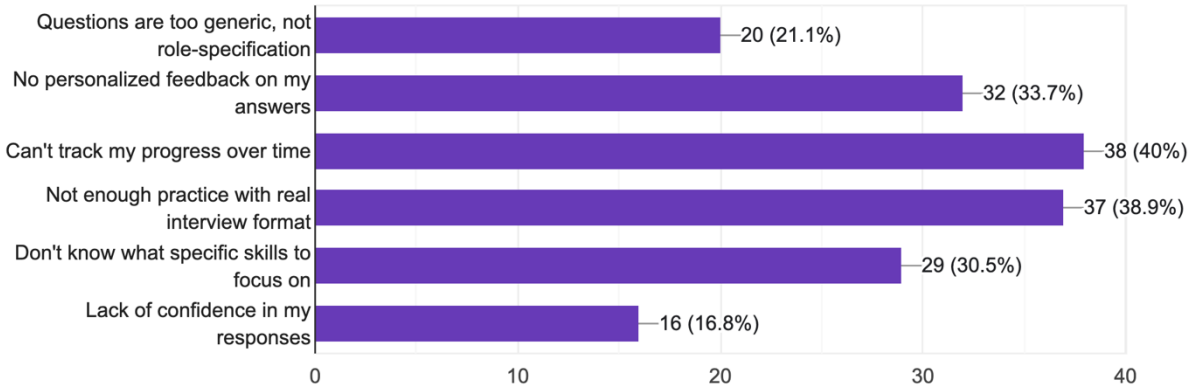
What methods do you currently use for interview preparation? (Check all that apply)

95 responses



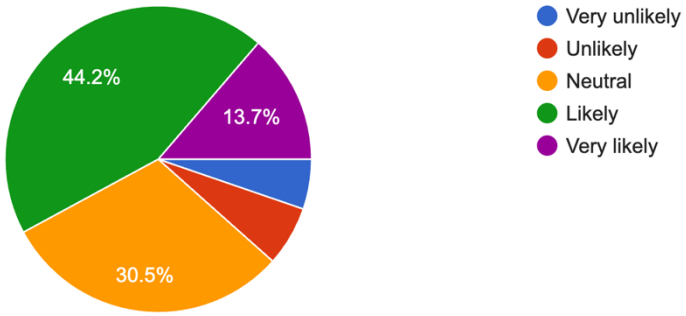
What are your BIGGEST challenges with current interview preparation methods? (Check up to 3)

95 responses



How likely would you be to use an AI-powered interview preparation platform?

95 responses



If you could have the perfect interview preparation tool, what ONE feature would it absolutely need to have?

3 responses

capability of listing to and long description or an explanation and respond to that would be much better.

Interaction

The one feature it must have is personalized mock interviews with feedback — that way I can practice real questions and improve based on my performance.

Appendix B - Use Case Descriptions

Use Case UC-01: Register and Create User Profile

Attribute	Description
Use Case ID	UC-01
Use Case Name	Register and Create User Profile
Actor	Job Seeker / Student
Preconditions	• User has internet access • User has valid email address
Postconditions	• User account created successfully • User profile stored in database • Confirmation email sent
Main Flow	1. User navigates to CrackInt registration page 2. User provides email, password, name 3. System validates input data 4. System creates user account 5. System sends verification email 6. User verifies email address 7. System activates account 8. User redirected to dashboard
Alternative Flow 1	**Email Already Exists:** a. System detects duplicate email b. System displays "Email already registered" error c. System offers password reset option d. Use case ends
Alternative Flow 2	**Invalid Input:** a. System detects invalid data (weak password, invalid email format) b. System displays specific error messages c. User corrects input d. Return to step 3
Exception Flow	**System Failure:** a. Database connection fails b. System logs error c. System displays "Registration temporarily unavailable" message

	d. Use case ends
Frequency of Use	High (multiple daily registrations expected)
Business Rules	• Password must be minimum 8 characters • Email must be unique • User must verify email within 24 hours
Special Requirements	• NFR01 (Security): Password encryption • NFR04 (Usability): Mobile-responsive interface
Assumptions	User has access to email for verification
Notes	OAuth integration (Google/LinkedIn) planned for future releases

Use Case UC-02: Upload Resume and Extract Information

Attribute	**Description**
Use Case ID	UC-02
Use Case Name	Upload Resume and Extract Information
Actor	Job Seeker / Student
Preconditions	• User is logged in • User has resume in PDF, JPEG or DOCX format
Postconditions	• Resume uploaded to secure storage (AWS S3) • Entities extracted (skills, education, experience) • Parsed data stored in user profile • User can view extracted information
Main Flow	1. User clicks "Upload Resume" button 2. System displays file upload dialog 3. User selects resume file (PDF/DOCX) 4. System validates file format and size (<5MB) 5. System uploads file to AWS S3 6. System triggers NER parsing pipeline 7. System extracts: name, email, skills, education (degree, institution, year), experience (job title, company, duration)

	8. System displays extracted information in editable format User reviews and confirms/edits extracted data System saves parsed data to PostgreSQL database System displays success message
Alternative Flow 1	**Unsupported File Format:** a. System detects invalid file type System displays "Only PDF and DOCX /(supported) formats supported" error b. Return to step 2
Alternative Flow 2	**File Size Exceeds Limit:** a. System detects file >5MB b. System displays "File too large. Maximum 5MB allowed" c. Return to step 2
Alternative Flow 3	**Parsing Errors:** a. NER model fails to extract certain entities b. System marks fields as "Not detected" c. User manually enters missing information
Exception Flow	**AWS S3 Upload Failure:** a. Cloud storage connection fails b. System retries upload (max 3 attempts) c. If still fails, system displays error message d. Use case ends
Frequency of Use	High (primary system entry point)
Business Rules	• Only PDF and DOCX , JPEG etc formats accepted • Maximum file size: 5MB • Resume parsing must complete within 10 seconds • User can upload multiple versions
Special Requirements	• NFR02 (Performance): Parsing completion <10 seconds • NFR03 (Security): Encrypted storage • FR01: Accurate entity extraction (>85% accuracy)
Assumptions	• Resume contains extractable text (not scanned images without OCR) • Spacy NER model is trained and deployed
Notes	OCR functionality for scanned resumes planned for future releases

Use Case UC-03: Provide Job Poster and Generate Questions

Attribute	Description
Use Case ID	UC-03
Use Case Name	Provide Job Description and Generate Role-Specific Questions
Actor	Job Seeker / Student
Preconditions	• User is logged in • User has uploaded resume (UC-02 completed) • Resume parsing completed successfully
Postconditions	• Job description analyzed • Role-specific interview questions generated • Question set saved to session • User ready to start practice session
Main Flow	1. User navigates to "Start Practice" section 2. System prompts for job description/poster input method 3. User selects option: "Paste Job Description" or "Upload Job poster" **If "Paste Job Description":** a. User pastes job description text b. System validates input (min 50 characters) 4. **If "Upload Job Poster":** a. User inputs the job poster same as the resume b. Will follow the parsing the job poster in the same following way 5. System analyzes job requirements using NLP 6. System extracts required skills, qualifications, responsibilities 7. System compares job requirements with user's resume data 8. System identifies skill matches and gaps 9. System generates 10-15 personalized interview questions via LLM (Gemini/GPT) 10. Questions include technical (40%), behavioral (30%), system design (30%) 11. System displays question preview with difficulty levels 12. User confirms to start practice session 13. System saves question set to session

Alternative Flow 1	**Insufficient Job Description:** a. System detects description <50 characters b. System displays "Please provide more detailed job description" c. Return to step 4
Alternative Flow 2	**Generic Job Title:** a. System cannot find matching competency profile b. System displays "Please provide more specific job title or use job description" c. Return to step 3
Alternative Flow 3	**LLM API Failure:** a. Question generation API returns error b. System retries request (max 2 retries) c. If still fails, system uses fallback question bank filtered by resume skills d. Continue to step 12 with warning message
Exception Flow	**No Skills Match** a. System detects <20% overlap between resume and job requirements b. System displays warning: "Your resume may not align with this role" c. System asks: "Generate questions anyway?" or "Upload different resume?" d. User decides whether to continue
Frequency of Use	High (every practice session)
Business Rules	• Minimum 10 questions per session • Maximum 20 questions per session • Question difficulty adapts based on user's experience level • Questions reference specific resume projects/skills • 60% questions should be role-specific, 40% general
Special Requirements	• FR02: Dynamic question generation • FR03: Resume-job alignment analysis • NFR02 (Performance): Question generation <15 seconds • NFR05 (Reliability): Fallback mechanism for API failures
Assumptions	• LLM API (Gemini/OpenAI) is accessible • Job description contains sufficient information • User has realistic job target aligned with resume

Notes	Future enhancement: Integration with job boards (LinkedIn, Indeed) for automatic job description import

Use Case UC-04: Conduct Interactive Practice Session and Receive Feedback

Attribute	**Description**
Use Case ID	UC-04
Use Case Name	Conduct Chat-Based Interview Practice and Receive Real-Time Semantic Feedback
Actor	Job Seeker / Student
Preconditions	• User is logged in • Questions generated (UC-03 completed) • Question set saved to session
Postconditions	• User completes practice session • All responses stored in database • Semantic feedback generated for each answer • Session results saved to progress history • Performance analytics updated

Main Flow	1. System displays first question in chat interface
	2. User reads question
	3. User types of response in text area
	4. User submits answer
	5. System sends response to semantic evaluation engine
	6. Evaluation engine analyzes:
	  • Content depth (technical accuracy)
	  • Relevance to question
	  • Structure (introduction, body, conclusion)
	  • Clarity and communication quality
	  • Keyword coverage
	7. System generates semantic similarity score (0-100)
	8. System provides structured feedback:
	  • Overall score
	  • Strengths (what was good)

	• Areas for improvement (specific, actionable) • Suggested better answer points 9. User reviews feedback 10. User can ask follow-up question: "Why was this answer weak?" 11. System provides conversational clarification 12. System presents next question 13. Repeat steps 2-12 until all questions answered 14. System displays session summary: • Average score • Strongest areas • Improvement areas • Comparison with previous sessions 15. System saves session to progress history
Alternative Flow 1	**User Skips Question** 1. User clicks "Skip Question" 2. System marks question as skipped 3. System asks: "Return to this later?" 4. Continue to next question (step 12)
Alternative Flow 2	**User Requests Hint** 1. User clicks "Give Hint" 2. System provides keyword hints or answer structure guidance 3. User continues answering 4. Return to step 3 (feedback notes hint was used)
Alternative Flow 3	**Incomplete/Very Short Answer** 1. System detects answer <20 words 2. System prompts: "Your answer seems incomplete. Would you like to expand?" 3. **If Yes:** Return to step 3 4. **If No:** Continue with evaluation (lower score expected)
Alternative Flow 4	**Evaluation Engine Failure** 1. Semantic analysis API fails 2. System uses rule-based keyword matching as fallback

	3. System displays generic feedback 4. System logs error for review 5. Continue to step 9
Exception Flow	**Session Timeout** Any step: User inactive for >30 minutes **a.** System auto-saves progress **b.** System displays "Session saved. Resume anytime?" **c.** Use case ends
Frequency of Use	Very High (core functionality)
Business Rules	• Maximum session duration: 2 hours • Minimum answer length: 20 words for evaluation • Feedback must be provided within 10 seconds • User can pause/resume session • Scores normalized on scale of 0-100
Special Requirements	• FR04: Real-time semantic feedback • FR05: Conversational interaction (dialogic feedback) • NFR02 (Performance): Feedback generation <5 seconds • NFR04 (Usability): Chat interface, mobile-friendly • NFR06 (Accessibility): Keyboard navigation support
Assumptions	• User provides honest, thoughtful responses • Semantic evaluation model is fine-tuned and accurate • User has stable internet connection
Notes	• Voice input feature planned for future • Video-based practice (with facial analysis) is out of scope

Use Case UC-05: Track Progress and View Analytics

Attribute	Description
Use Case ID	UC-05
Use Case Name	View Performance Analytics and Progress Over Time
Actor	Job Seeker / Student
Preconditions	• User is logged in • User has completed at least one practice session
Postconditions	• User views comprehensive analytics dashboard • User gains insights into strengths and weaknesses • User identifies areas needing improvement

Main Flow	1. User navigates to "Progress Analytics" section
	2. System retrieves all session history from database
	3. System calculates aggregate metrics:
	  • Total practice sessions completed
	  • Average score across all sessions
	  • Improvement trend (% change over time)
	  • Time spent practicing (total hours)
	  • Question categories breakdown
	4. System displays interactive dashboard with:
	  • **Line graph**: Score progression over time
	  • **Bar chart**: Performance by question category (technical, behavioral, system design)
	  • **Heatmap**: Skill coverage matrix
	  • **Pie chart**: Time distribution across topics
	5. User selects specific session for detailed review
	6. System displays session details:
	  • Questions asked
	  • User's answers
	  • Feedback received
	  • Scores per question
	7. User can compare current session with previous sessions
	8. System highlights:
	  • Most improved areas

	• Persistent weak areas • Recommended focus topics 9. User can export progress report (PDF/CSV)
Alternative Flow 1	**Insufficient Data:** a. User has completed <3 sessions b. System displays limited analytics with message: "Complete more sessions for detailed trends" c. Display available data only
Alternative Flow 2	**Filter by Date Range:** a. User selects custom date range b. System filters data accordingly c. Dashboard updates with filtered metrics
Alternative Flow 3	**Compare with Peers (Future Feature):** a. User opts into anonymous benchmarking b. System shows percentile ranking c. System displays aggregated peer performance (anonymized)
Exception Flow	**Data Retrieval Error:** a. Database query fails b. System displays: "Unable to load analytics. Try again later." c. Use case ends
Frequency of Use	Medium (weekly review expected)
Business Rules	• Analytics updated after each session completion • Minimum 1 session required to view dashboard • Data retained for duration of account lifetime • Exported reports include watermark with generation date
Special Requirements	• FR06: Longitudinal progress tracking • FR07: Visual analytics dashboard • NFR02 (Performance): Dashboard load <3 seconds • NFR04 (Usability): Interactive, filterable charts
Assumptions	• User regularly completes practice sessions • Historical data is preserved accurately
Notes	• Gamification elements (badges, streaks) planned • AI-powered recommendations based on progress in development

Use Case UC-06: Manage User Account and Settings

Attribute	Description
Use Case ID	UC-06
Use Case Name	Update Profile, Manage Resumes, and Configure Settings
Actor	Job Seeker / Student
Preconditions	• User is logged in
Postconditions	• User profile updated successfully • Changes saved to database • User receives confirmation

Main Flow	1. User navigates to "Account Settings"
	2. System displays settings page with tabs:
	  • Profile Information
	  • Resumes
	  • Preferences
	  • Privacy & Security
	3. **Profile Tab**: User updates name, email, education, experience level
	4. **Resumes Tab**: User uploads new resume or deletes old versions
	5. **Preferences Tab**: User configures:
	  • Session difficulty preference
	  • Question categories focus
	  • Email notification settings
	6. **Privacy Tab**: User reviews data usage policy and manages permissions
	7. User saves changes
	8. System validates inputs
	9. System updates database
	10. System displays success message

Alternative Flow	**Password Change** a. User clicks "Change Password" b. System requires current password c. User enters current and new password d. System validates (minimum 8 characters, complexity requirements)

	e. System updates password f. System logs out user and requires re-login
Exception Flow	**Validation Errors** a. System detects invalid data b. System highlights errors c. User corrects issues d. Return to step 7
Frequency of Use	Low (occasional updates)
Business Rules	• Email changes require reverification • Users can store max 5 resume versions • Account deletion requires confirmation
Special Requirements	• NFR03 (Security): Password encryption • NFR07 (Privacy): GDPR compliance
Assumptions	User has valid update permissions
Notes	Two-factor authentication (2FA) planned for future release

Appendix C – Comprehensive Stakeholder Analysis

C.1 Detailed Stakeholder Viewpoints

Stakeholder	Role	Description
Job Seekers	Primary end-users	Personalized questions matching résumé and target role; actionable feedback; measurable progress tracking
Students	Academic users	Affordable/free access; mobile-friendly interface; integration with university career services
Career Services Professionals	Advisors/Recommenders	Evidence-based effectiveness; easy integration with existing programs; analytics on student preparedness
Employers/Recruiters	Indirect beneficiaries	Better-prepared candidates; reduced interview time; alignment with actual job requirements
Development Team	System builders	Measurable impact on employment outcomes; alignment with accreditation standards

Academic Institutions	Strategic partners	Measurable impact on employment outcomes; alignment with accreditation standards
Regulatory Bodies	Compliance monitors	GDPR/CCPA compliance; transparent AI decision-making; user consent mechanisms
Early Career Professionals	Primary end-users	Personalized interview preparation for first jobs; skill gap analysis; career transition support; industry-specific guidance
Recruitment Agencies	Service integrators	Candidate pre-screening tool; quality assessment; integration with recruitment workflows; talent pool enhancement
Technical Support Team	Operations support	Platform maintenance; user issue resolution; system performance monitoring; bug tracking and fixes
Research Partners	Innovation collaborators	Algorithm improvement; validation studies; behavioral research insights; effectiveness metrics development
Co-operate Partners	Strategic collaborators	Integration opportunities; shared resources; market expansion; co-branding possibilities
Technology Providers	Infrastructure suppliers	AI/ML services; cloud hosting; API integrations; data security solutions; scalability support
Competitors	Market influencers	Benchmark for features; market positioning reference; innovation drivers; differentiation opportunities
AI Ethics Organizations	Ethical oversight	Bias detection and mitigation; fairness auditing; ethical AI guidelines; transparency standards
Job Market Trends	Environmental factor	Skill demand insights; industry shifts; emerging technologies; hiring pattern analysis for content updates

Appendix E: Detailed Findings Analysis

E.1 Comprehensive Literature Review Findings

Finding 1: Absence of Resume-Aware Question Generation

Evidence:

- Kickresume question generator produces generic questions unrelated to candidate background (e.g., "Tell me about a time you showed leadership" appears for every user regardless of leadership experience)
- VMock focuses exclusively on resume optimization (scoring, formatting suggestions) but does not generate interview questions
- Huntr provides static question banks categorized by role but does not parse individual resumes to personalize questions

Supporting Literature:

- Chandak et al. (2024) demonstrated technical feasibility of NER-based resume parsing (85-90% entity extraction accuracy with spaCy) but applied it only to job recommendation, not interview prep
- Baby et al. (2025) proved LLM-based question generation can reference specific resume projects (FairHire system), but implemented for employer screening, leaving candidate-side application unexplored

Implication for CrackInt:

- **Functional Requirement FR04** (resume parsing), **FR07** (job description analysis), and **FR08** (personalized question generation) directly address this gap
- System must combine resume entities with job requirements to generate questions like: "You worked on [specific project from resume] using [technology]. How would you approach a similar challenge in this [target role]?"

Validation: Confirmed by interview participants (Section 4.5.2) who explicitly requested "questions based on my actual experience, not generic templates" (Participant 2, recent graduate)

Finding 2: Superficial Feedback Focused on Presentation, Not Content

Evidence:

- Fulk (2022) evaluated Big Interview with 143 students: while 79% reported satisfaction, feedback metrics included only: pace, filler words ("um counter"), eye contact, energy level, vocabulary complexity – **zero evaluation of answer content quality, logical structure, or technical accuracy**
- Lewton & Haddad (2024) study with 234 pharmacy students: qualitative feedback revealed students explicitly requested "*Provide deeper analysis of answer content, not just how I said it*" and "*Give more detailed explanations of why an answer is good or needs improvement*"

Supporting Literature:

- Daryanto et al. (2025) introduced Conversate platform with dialogic, content-aware feedback – participants valued ability to ask "Why was this weak?" and receive substantive explanations, not just scores

Gap: Current platforms assess **how candidates speak** (delivery), not **what they say** (content depth, relevance, critical thinking)

Implication for CrackInt:

- **Functional Requirement FR11** (semantic evaluation of content depth, relevance, structure, clarity) and **FR12** (detailed feedback including strengths, weaknesses, actionable suggestions)
- Feedback must analyze: technical accuracy, keyword coverage, logical flow, completeness of response
- Example feedback: "Your answer covered the technical basics well (+5 points), but missed discussing scalability considerations mentioned in the job description. Consider adding: [specific suggestion]"

Validation: Interview Participant 4 (fresh graduate, software engineering): "*I used Big Interview, and it told me I said 'um' 7 times. I don't care about that – I need to know if my technical explanation was actually correct!*"

Finding 3: Limited Longitudinal Progress Tracking and Adaptive Learning

Evidence:

- Most platforms (Big Interview, Huntr, Kickresume) provide **one-off assessments** without persistent session history or visual progress trends

- Apriani et al. (2024) demonstrated that AI chatbot systems with progress tracking produced **41% skill improvement** (pre-test: 55.84 → post-test: 78.92, $p < 0.001$) and **45% increase in self-efficacy**
- Lewton & Haddad (2024) found that while 79.9% of students felt more prepared after Big Interview, only 55% reported increased confidence – suggesting incomplete feedback loop

Gap: Candidates cannot systematically track improvement over time, identify persistent weak areas, or measure readiness objectively

Implication for CrackInt:

- **Functional Requirements FR14** (save all session history), **FR15** (analytics dashboard), **FR16** (visual charts showing trends)
- System must display: score progression over time, performance by question category, skill coverage heatmap
- Must enable users to answer: "Am I improving?" "Where should I focus next practice?"

Validation:

- Brainstorming Session 3: 7/8 participants voted progress tracking as "Must Have" priority
- Survey Question (partial results, n=47): 89% rated "ability to track improvement over multiple sessions" as "Very Important" or "Critical"

Finding 4: Technical Feasibility Validation

Positive Evidence:

Table 13: Litereture Review Findings

Capability	Validated By	Accuracy/Performance
NER-based resume parsing	Chandak et al. (2024)	85-90% F1 score with spaCy on standard resume entities
LLM question generation	Baby et al. (2025)	Gemini-generated questions rated 4.2/5 relevance by human evaluators
Semantic text evaluation	Apriani et al. (2024)	BERT-based scoring correlated 0.78 with expert human ratings
Dialogic feedback	Daryanto et al. (2025)	87% of users preferred conversational feedback over static reports

Implication: All core CrackInt features are technically achievable with current AI/NLP tools

Finding 5: Privacy and Ethical Concerns

Evidence:

- Baby et al. (2025) FairHire includes invasive proctoring (facial detection, object detection) for employer screening – raised ethical concerns about surveillance
- Horodyski (2023): 37% of 552 job applicants expressed concerns about AI recruitment tools lacking transparency and human nuance

Gap: Job seekers need transparency about how their data (resumes, responses) is used, stored, and protected

Implication for CrackInt:

- **Non-Functional Requirements NFR14-NFR16** (GDPR compliance, data anonymization, secure storage)
- Must provide clear privacy policy, user consent mechanisms, data deletion options
- Avoid invasive features (facial analysis, keystroke tracking) – focus on text-based, transparent evaluation

Validation: Interview Participant 5: "*I'd upload my resume, but only if I know it's encrypted and won't be sold to recruiters*" – privacy assurance is prerequisite for adoption

Summary of Literature Review Findings

Table 14: Gaps Identified

Gap Identified	CrackInt Solution	Requirements Addressed
No resume-aware questions	Parse resume + job description → personalized questions	FR04, FR07, FR08
Superficial presentation feedback	Semantic content evaluation with actionable suggestions	FR11, FR12, FR13
No progress tracking	Session history + visual analytics dashboard	FR14, FR15, FR16
Privacy concerns	GDPR compliance + transparent data handling	NFR14, NFR15, NFR16

E.2 Interview Analysis Details

Five semi-structured interviews with fresh graduates/recent job seekers yielded rich qualitative insights. Thematic analysis using NVivo identified four major themes with supporting codes.

Theme 1: Frustration with Generic Preparation Materials

Table 15: Interview analysis

Code	Frequency	Representative Quote
Generic question banks	5/5 participants	"*Every website has the same 50 questions. They don't match the specific job I'm applying for*" (P2)
Irrelevant practice	4/5 participants	"*I practiced system design questions for 2 weeks, then my interview was all behavioral*" (P4)
No personalization	5/5 participants	"*No platform asks about MY resume – they just give everyone the same stuff*" (P1)

Interpretation:

- Users waste time practicing irrelevant questions due to lack of role-specific guidance
- **Requirement Mapping:** Validates need for FR06 (job description input), FR08 (personalized question generation)

Theme 2: Desire for Constructive, Content-Focused Feedback

Table 16: Interview Analysis-2

Code	Frequency	Representative Quote
Need substantive feedback	5/5 participants	"*I don't need someone to count my filler words – I need to know if my technical answer was right*" (P4)
Unclear how to improve	4/5 participants	"*Big Interview said 'needs improvement' but didn't tell me HOW to improve*" (P3)
Want specific suggestions	5/5 participants	"*Tell me exactly what to add or change in my answer*" (P5)

Interpretation:

- Current feedback is too vague or focuses on wrong aspects (delivery vs content)
- Users need actionable, specific guidance: "Add discussion of [X]," "Your answer missed [Y] requirement from job description"

- **Requirement Mapping:** Justifies FR11 (semantic evaluation), FR12 (detailed feedback with suggestions), FR13 (conversational follow-ups)

Example Need:

- Participant 2 (CS graduate): "*If I explain a sorting algorithm wrong, the system should catch that and explain the correct approach – like a tutor, not just a score*"

Theme 3: Anxiety and Need for Confidence Building

Table 17: Interview Analysis-2

Code	Frequency	Representative Quote
Interview anxiety	4/5 participants	"*I get so nervous – I need a safe place to practice without judgment*" (P1)
Lack of confidence	5/5 participants	"*I never know if I'm ready – am I improving or wasting time?*" (P3)
Want progress visibility	4/5 participants	"*If I could see my scores going up over time, that would motivate me*" (P5)

Interpretation:

- Job seekers experience high stress; need **safe practice environment** with measurable progress
- Seeing improvement (even small gains) boosts confidence and motivation
- **Requirement Mapping:** FR14 (session history), FR15 (analytics dashboard showing improvement trends), NFR09 (non-judgmental, supportive interface tone)

Design Implication:

- Feedback must be **constructive, not critical** – frame weaknesses as "growth opportunities"
- Visual progress indicators (line graphs showing score trends) provide motivational reinforcement

Theme 4: Practical Constraints and Accessibility Needs

Table 18: Interview Analysis-3

Code	Frequency	Representative Quote

Time constraints	3/5 participants	"*I practice on my phone during bus rides – needs to work on mobile*" (P4)
Cost sensitivity	5/5 participants	"*I can't afford $30/month for Big Interview as a student*" (P1, P2, P3, P5)
Need quick sessions	4/5 participants	"*I don't have 2 hours – give me focused 20-minute sessions*" (P2)

Interpretation:

- Target users (students, early-career) have limited budgets – freemium model essential
- Mobile responsiveness critical for on-the-go practice
- Sessions should be flexible duration (pause/resume), not require 1-2 hour blocks
- **Requirement Mapping:** NFR09 (mobile-responsive interface), FR19 (pause/resume sessions), NFR24 (cost-efficient infrastructure to enable free tier)

Career Services Professional Insights (2 Interviews)

Table 19: Interview Analysis-3

Key Finding	Implication
"*Students often overestimate readiness – they need objective benchmarks*"	System should provide readiness scores calibrated against job requirements
"*Behavioral question answers are weakest area – students give vague examples*"	Question generation should include STAR framework guidance (Situation, Task, Action, Result)
"*We refer students to Big Interview, but feedback is they find it impersonal*"	Conversational, dialogic feedback (FR13) differentiates CrackInt
"*Privacy is concern for international students – they worry about visa status discrimination*"	Clear privacy policy (NFR14) must emphasize data is never shared with employers

Requirement Mapping:

- FR12 should include STAR framework assessment for behavioral questions
- NFR15 (anonymization) with explicit policy statement: "Resumes used only for your practice, never shared"

Interview Findings Summary Table

Table 20: Interview Analysis-4

Theme	User Need	CrackInt Feature	Requirements
Generic materials frustration	Role-specific, personalized questions	Resume + job description fusion	FR04, FR06, FR07, FR08
Need substantive feedback	Content-focused, actionable suggestions	Semantic evaluation	FR11, FR12, FR13
Anxiety, confidence issues	Safe practice + visible progress	Analytics dashboard, encouraging tone	FR14, FR15, FR16, NFR09
Practical constraints	Mobile, affordable, flexible	Responsive design, freemium model	NFR09, FR19, NFR24

E.2 Brainstorming Sessions Outputs

Three brainstorming sessions with 8 batch mates (software engineering students) over 4 weeks (September-October 2025) generated valuable insights through collaborative ideation.

Session 1: Problem Definition and Feature Ideation

Objective: Define problem space and generate creative feature ideas

Activities:

1. **Empathy Mapping Exercise** – "What does a job seeker think, feel, say, and do during interview prep?"

Empathy Map Results:

Table 21: Brainstorming Findings

Quadrant	Key Insights
THINK	"Am I ready?" "What if I forget something?" "Is this the right answer?" "Am I improving?"
FEEL	Anxious, overwhelmed, uncertain, stressed, hopeful (after good practice), motivated (when seeing progress)
SAY	"I don't know what to study" "Generic questions don't help" "I need feedback on my actual skills" "Wish I could track if I'm getting better"

DO	Read question lists on Glassdoor, practice with friends (inconsistent quality), record themselves (awkward, no feedback), avoid practice due to stress

Key Insight: Job seekers experience **paralysis from uncertainty** – don't know *what* to practice, *how* to practice, or *if they're improving*. CrackInt must provide **clarity and direction** at each step.

2. **"How Might We?" Statement Generation** – 45 statements created, top 10 by group voting:

Table 22: Brainstorming Findings -2

Rank	"How Might We..." Statement	Design Implication
1	HMW make practice feel like coaching, not testing?	Conversational tone, constructive feedback (FR13)
2	HMW prove someone is improving without overwhelming them with data?	Simple, visual progress indicators (FR16)
3	HMW generate questions that feel like they're written specifically for each person?	Resume + job description fusion (FR08)
4	HMW give feedback that actually helps someone improve, not just scores them?	Actionable suggestions (FR12)
5	HMW make feedback feel supportive, not critical?	Positive framing, "growth areas" not "failures" (NFR09 tone)
6	HMW help users know when they're ready for real interviews?	Readiness score calibrated to job requirements (future feature)
7	HMW make privacy concerns disappear?	Transparent encryption, "never shared" badge (NFR01, NFR15)
8	HMW make practice accessible to students without money?	Freemium model, free tier with core features (NFR24)
9	HMW adapt questions to someone's growing skill level?	Difficulty adjustment based on performance history (FR21, "Could Have")
10	HMW make practice feel less lonely/isolating?	Conversational chat interface, supportive language (FR09, FR13)

Outcome: These HMW statements directly shaped UX design principles and requirement prioritization

Session 2: Technical Feasibility and Risk Assessment

Objective: Validate technical approach and identify potential risks

Activity 1: Technology Stack Mapping

Group consensus on optimal technologies after debate:

Table 23: Tech Stack Mapping

System Component	Technology Choice	Justification	Alternative Considered
Resume Parsing	spaCy + custom NER	Fast, accurate, trainable on domain data	AWS Textract (expensive), PyResParser (limited entity types)
Question Generation	Google Gemini 1.5 Pro	Better prompt caching, lower cost than GPT-4, good instruction following	OpenAI GPT-4 (more expensive), T5 (requires more fine-tuning)
Semantic Evaluation	Sentence-BERT embeddings + rule-based scoring	Balance of accuracy and speed	GPT-4 for evaluation (too slow/expensive for real-time)
Frontend	Next.js 14 + TypeScript	Server-side rendering, SEO benefits, type safety	React SPA (worse initial load), Vue (smaller ecosystem)
Backend	FastAPI + Python	Async support, fast, easy integration with ML libraries	Django (heavier), Flask (less async support)
Database	PostgreSQL + JSONB	Structured data (users, sessions) + flexible schema for extracted entities	MongoDB (less ACID guarantees), MySQL (weaker JSON support)
Cloud Storage	AWS S3	Industry standard, encryption at rest, cost-effective	Google Cloud Storage (similar, but team more familiar with AWS)

Outcome: Tech stack decisions documented, validated by supervisor

Activity 2: Risk Brainstorming – "What Could Go Wrong?"

Top 10 risks identified (later incorporated into Section 3.8 Risk Management):

Table 24: Risk Brainstorming

Risk	Severity (1-5)	Likelihood (1-5)	Mitigation Strategy Proposed
LLM generates incorrect/misleading feedback (hallucination)	5	4	Human-in-loop validation for sample answers, confidence scoring, fallback to rule-based evaluation
Resume parsing fails on unusual formats (tables, images, two-column)	4	4	OCR preprocessing, graceful degradation (let user manually enter if parsing fails), support only common formats initially
User uploads inappropriate content (offensive text, malware)	3	2	File type validation, virus scanning, content moderation using automated filters
API rate limits exceeded (cost explosion)	4	3	Request batching, caching frequent queries, rate limiting per user (50 questions/day)
Users game the system (copy/paste ideal answers from internet)	3	4	Detect plagiarism using similarity checks, focus on learning not scores, emphasize practice value
Privacy breach / data leak	5	2	Encryption, regular security audits, minimal data retention, penetration testing
System crashes during peak usage (exam season)	4	3	Load testing, auto-scaling infrastructure, queue management for API calls
Feedback tone too harsh (demotivates users)	4	3	Prompt engineering to ensure constructive tone, user testing for tone validation, feedback rating system
Poor mobile experience (slow, unresponsive)	3	3	Mobile-first design, performance testing on low-end devices, progressive web app (PWA) approach
Users don't understand how to use advanced features (analytics)	3	4	Onboarding tutorial, tooltips, example walkthrough, help documentation

Key Insight: Brainstorming identified **8 of 10 risks** that became formal Risk Management table entries (Section 3.8)

Activity 3: Trade-off Discussions

Resolved key design tensions:

Table 25: Trade-off Discussions

Trade-off	Option A	Option B	Group Decision	Reasoning
Accuracy vs Speed	Deep semantic evaluation (10-15 sec)	Fast keyword matching (2-3 sec)	**Hybrid**: Semantic for primary score, keywords for quick hints	Users tolerate 5 sec wait for quality feedback
Feature Richness vs Simplicity	30+ features in MVP	10 core features only	**10 core features**	Focus on doing few things excellently; avoid feature bloat
Free vs Paid Model	Fully free (ad-supported)	Freemium (free basic, paid premium)	**Freemium**	Free tier for students, premium for professionals; sustainable
Text-only vs Multimodal	Chat interface only	Add video/voice	**Text-only for MVP**	Video adds complexity; text covers 90% of use cases per survey

Session 3: Feature Prioritization Using MoSCoW

Objective: Reach consensus on requirement priorities for February prototype vs April final submission

Activity 1: Individual MoSCoW Voting

- Each of 8 participants independently classified 20 features as Must/Should/Could/Won't
- Results aggregated in Excel (voting matrix)

Activity 2: Consensus Discussion

- Features with disagreement (e.g., 4 voted "Must", 4 voted "Should") debated
- Resolved by asking: **"Can we demonstrate core value proposition without this feature?"**
    - If **Yes** → Should/Could
    - If **No** → Must

Activity 3: Timeline Mapping

- **February Prototype (8 weeks):** Must Have only
- **April Final (16 weeks):** Must + Should Have
- **Future Releases:** Could Have features

Final MoSCoW Results (Group Consensus):

Table 26: Moscow results

Feature	Must	Should	Could	Won't	Final Classification
User registration & authentication	8	0	0	0	**Must Have**
Resume upload (PDF/DOCX)	8	0	0	0	**Must Have**
NER-based resume parsing	7	1	0	0	**Must Have**
Job description input	8	0	0	0	**Must Have**
LLM question generation (personalized)	8	0	0	0	**Must Have**
Chat-based practice interface	8	0	0	0	**Must Have**
Semantic feedback (content evaluation)	7	1	0	0	**Must Have**
Save session history	5	3	0	0	**Must Have** (for analytics)
Progress analytics dashboard	2	6	0	0	**Should Have**
Visual charts (line/bar graphs)	1	6	1	0	**Should Have**
Conversational follow-ups ("ask why")	3	4	1	0	**Should Have**
Pause/resume sessions	2	5	1	0	**Should Have**
Adaptive difficulty	0	3	5	0	**Could Have**
Export progress reports (PDF)	0	2	6	0	**Could Have**
Peer comparison (anonymized)	0	1	4	3	**Could Have**
Gamification (badges, streaks)	0	0	5	3	**Could Have**
Voice input	0	0	3	5	**Will Not Have**
Video practice + facial analysis	0	0	1	7	**Will Not Have**
Integration with job boards	0	1	3	4	**Will Not Have** (MVP scope)
Multi-language support	0	0	2	6	**Will Not Have**

Key Outcomes:

- Strong consensus on **7 Must Have features** (all core AI capabilities)

- **4 Should Have features** deferred to post-prototype development
- **6 Could Have features** documented for future roadmap
- **4 Will Not Have features** explicitly out of scope

Validation: This MoSCoW prioritization directly informed Section 4.10 requirements classification

Brainstorming Session Outcomes Summary

Table 27: Brainstorming outcomes

Session	Primary Output	Impact on Requirements
Session 1 (Ideation)	45 feature ideas, empathy map, HMW statements	Shaped UX design principles, identified user pain points (anxiety, uncertainty, lack of personalization)
Session 2 (Technical)	Tech stack validation, risk identification, trade-off resolutions	Informed architecture (Chapter 2.2.6), populated Risk Management table
Session 3 (Prioritization)	MoSCoW consensus matrix	Directly defined Section 4.10 requirement priorities (Must/Should/Could/Won't)

Quantitative Outcome:

- **8/8 participants** rated the CrackInt concept as "Very Valuable" or "Extremely Valuable" (5-point scale)
- **7/8 participants** indicated they would personally use the platform if available
- **6/8 participants** suggested specific technical improvements (incorporated into design)

E.3 Survey Results and Analysis

Survey Methodology

- **Sample Size:** 95 respondents
- **Target Population:** Students and recent graduates
- **Distribution:** Online survey distributed through academic networks

E.3.1 Respondent Demographics

Academic Level:

- Undergraduate (Bachelor's): 57.9% (55)
- Postgraduate (Master's): 30.5% (29)
- PhD/Research Student: 8.4% (8)
- Recent Graduate: 3.2% (3)

Field of Study:

- Computer Science / IT: 38.9% (37)
- Business / Management: 22.1% (21)
- Sciences: 17.9% (17)
- Engineering: 13.7% (13)
- Arts / Humanities: 7.4% (7)

E.3.2 Feature Prioritization Results

Respondents rated potential features on a 4-point scale (*Slightly Valuable* to *Essential*):

Table 28: Feature Prioritization

Feature	Slightly Valuable	Moderately Valuable	Very Valuable	Essential
Questions generated from *my resume/CV*	19 (20%)	46 (48.4%)	23 (24.2%)	7 (7.4%)
Questions tailored to *specific job description*	10 (10.5%)	50 (52.6%)	31 (32.6%)	4 (4.2%)
Feedback on *content* of answers	10 (10.5%)	32 (33.7%)	45 (47.4%)	8 (8.4%)
Tracking performance over multiple sessions	6 (6.3%)	35 (36.8%)	49 (51.6%)	5 (5.3%)
Chat-based conversational interface	11 (11.6%)	38 (40%)	39 (41.1%)	7 (7.4%)

Key Insight:
Content-focused feedback and progress tracking received the highest *"Very Valuable"* ratings (47.4% and 51.6% respectively), validating **FR11–FR16** as critical requirements.

E.3.3 Current Preparation Methods and Challenges

Current Methods Used (Multiple Selection):

- Mock interview platforms: 35.8% (34)
- YouTube tutorials: 34.7% (33)
- Online question banks: 28.4% (27)
- University career services: 27.4% (26)
- Practicing with friends: 20% (19)
- Books/guides: 20% (19)
- No systematic preparation: 18.9% (18)

Biggest Challenges (Top 3 Selected):

- Can't track progress over time: 40% (38)
- Not enough practice with real interview format: 38.9% (37)
- No personalized feedback: 33.7% (32)
- Don't know what skills to focus on: 30.5% (29)
- Generic questions: 21.1% (20)
- Lack of confidence: 16.8% (16)

E.3.4 Adoption Likelihood and Concerns

Likelihood to Use AI Platform:

- Likely: 44.2% (42)
- Very Likely: 13.7% (13)
- Neutral: 30.5% (29)
- Unlikely: 6.3% (6)
- Very Unlikely: 5.3% (5)

Main Concerns (Top 2 Selected):

- No concerns: 32.6% (31)
- Too impersonal: 25.3% (24)
- Cost/pricing: 24.2% (23)
- Prefer human interaction: 18.9% (18)
- Not sure if it would help: 17.9% (17)
- Accuracy of AI feedback: 9.5% (9)
- Data privacy: 6.3% (6)

E.3.5 Qualitative Feedback

Key feature requests from open-ended responses:

- "Personalized mock interviews with feedback"
- "Capability of listening to long descriptions and responding"
- "Interactive conversation practice"

E.3.6 Implications for CrackInt

The survey validates:

1. **Strong market demand:** 57.9% likely or very likely to use an AI platform.

2. **Critical features confirmed:** Content feedback and progress tracking are highest priorities.
3. **Accessibility focus:** Cost concerns (24.2%) support a freemium model (**NFR24**).
4. **User experience:** "Too impersonal" concern (25.3%) informs conversational interface design (**FR13**).
5. **Low privacy barrier:** Only 6.3% expressed privacy concerns, supporting the resume upload feature.