Milestone -1

App Name: On-Board

Project Description: On-Board is a web platform designed to help students, professionals, and career changers navigate their career paths. It offers personalized features like profile creation, job matching, and connections to alumni and industry leaders. The platform also includes an exceptional grouping feature, allowing users to join or create groups based on tags like industry, skills or job roles, enhancing collaboration and networking. On-Board provides a comprehensive solution for efficient career development.

Personas and User stories

On-Board is a platform designed to serve a wide range of users, primarily college students, early-career professionals, career changers, recruiters, mentors, and companies. College students, typically aged 18-24, benefit from the platform as they seek internships or firsttime job opportunities. Additionally, early-career professionals aged 25-35 use the platform to transition into new roles, network, and enhance their skills, while career changers find guidance and opportunities for reskilling. For recruiters and companies, On-Board offers a way to connect with fresh talent, advertise achievements, and promote job openings, while mentors and alumni provide career advice to younger generations.

The core problem On-Board addresses is the fragmentation of career resources and lack of networking opportunities, especially for students and career changers. Many individuals struggle to navigate the job market without proper guidance, face limited access to professional networks, and have difficulty managing job applications. On-Board consolidates multiple tools to help users gain personalized career advice, find mentorship, network effectively, and manage their job applications more efficiently. It also helps users identify skill gaps and suggests ways to develop them. The problem is significant, with over 40% of college graduates feeling underprepared for the workforce. Networking is crucial, with 70% of jobs filled through connections, yet many young professionals lack access to such networks. Job application tracking can be overwhelming, often leading to missed opportunities, with 40-60% of applications going unmonitored after submission. Moreover, mid-career professionals face challenges in switching industries due to the need for upskilling and reskilling, adding to the demand for a platform that facilitates smooth transitions.

The platform introduces several new features that differentiate it from others in the market. Tailored career recommendations, based on user profiles and industry trends, help users make informed career choices. The grouping feature fosters collaboration and peer support, while real-time job alerts keep users updated on relevant job opportunities. Mentorship connections bridge the gap between students and experienced professionals, and job application tracking ensures users stay organized throughout their job search. These features make On-Board a comprehensive platform for career growth and networking, offering solutions to both students and professionals navigating their career paths.

Milestone -1

Data Definition

User Data:

Attributes

- FirstName: First name of the user
- lastName: Last name of the user
- email: Email Address
- password: User's password (hashed for security)
- userType: Student/Recruiter/Company
- userCountry: Country in which the user resides or wants to work
- userState: State in which the user resides or wants to work
- userCity: City in which the user resides or wants to work
- userJobInterests: Job categories or roles that the user is interested in
- userRemote: Whether the user prefers remote jobs - userResume: Uploaded resume of the user (optional) - userSkills: List of skills the user possesses.
- userExperience: User's work experience (title, company, years)
- userEducation: User's educational background (degree, institution, years)
- userJobBookmarks: Jobs the user has bookmarked or saved
- userNotification: Job alerts, messages, and group notifications received by the user
- userTags: Keywords or tags for representing the user's areas of expertise

Student Data:

- studentId: Unique identifier for the student user.
- firstName: First name of the student.
- lastName: Last name of the student.
- Email: Email address of the student.
- Password: Hashed password for authentication.
- schoolName: Name of the school or university the student is attending.
- Degree. The degree or program the student is pursuing.
- Major the student's major or area of study.
- graduationYear: Expected year of graduation.
- studentSkills: List of skills acquired by the student (academic, technical, etc.) - studentResume Uploaded resume of the student.
- studentProjects List of projects completed by the student (title, description, technologies used).
- studentInternships: Details about internships or part-time.
- studentJobInterests: Job roles or industries the student is interested in.
- studentRemotePreference: Whether the student is interested in remote job.

- studentJobApplications: List of jobs the student has applied to.
- studentClubs: List of student clubs or extracurriculars
- studentAchievements Academic or extracurricular achievements.
- studentMentors : List of mentors the student relates to on the platform.
- studentNetworkingStatus: Student's engagement in networking activities.
- studentNotifications: Alerts for new jobs, messages, application status, etc. -
- studentProfileImage: Profile picture uploaded by the student.

Company Data:

Attributes

- companyName: Name of the company
- companyEmail: Official email address of the company
- companyWebsite: URL of the company's official website -
- companyIndustry: Industry in which the company operates -
- companyLocation: Physical location of the company.
- companySize: Number of employees in the company.
- companyDescription: Brief description of the company's mission, values, and services
- companyProducts: Products of the company
- jobOpenings: List of current job openings available at the company
- companyLogo: Logo image of the company
- companyTags: Keywords or tags representing the company's areas of expertise.

Recruiters Data

Attributes

- recruiterName: Full name of the recruiter
- recruiterEmail: Official email address of the recruiter
- recruiterPhone: Phone number of the recruiter
- recruiterCompany: Name of the company the recruiter represents
- recruiterLocation: Geographical location of the recruiter (city, state, country)
- recruiterSpecialty: Specific industries or job roles the recruiter specializes in
- jobOpenings: List of current job openings the recruiter is managing
- recruiterProfileLink: Link to the recruiter's professional profile (LinkedIn, etc.)

- recruiterTags: Keywords or tags related to the recruiter's areas of focus
- recruiterProfileImage: Profile picture of the recruiter

Group Data

Attributes

- groupId: Unique identifier for the group.
- groupName: Name of the group.
- groupDescription: Brief description of the group's purpose or focus.
- groupCreatorId: User ID of the group creator.
- groupMembers: List of user IDs who are members of the group.
- groupTags: Keywords or tags associated with the group (e.g., industry, roles).
- groupCreationDate: Date and time when the group was created.
- groupPosts: List of posts made within the group (post ID, author, timestamp).
- groupMessages: Conversations or messages exchanged between group members.
- groupReacts: Reactions (e.g., likes, comments) to posts within the group.
- groupJobsPosted: Job postings shared within the group (job ID, title, author).
- groupActiveStatus: Boolean indicating if the group is active or inactive.
- groupConversations: List of stored chat threads or conversations within the group.
- groupMatchConnections: Connections made between users within the group (matches).
- groupNotifications: Notifications related to group activities (new posts, messages).

Functional Requirements

User:

High Priority

1. User can register on the platform using their email and password.
2. Users must be able to securely log in and log out, with their passwords hashed for security.
3. Users must be able to recover or reset their passwords.
4. User can create Profile.
5. Users must be able to select their userType (e.g., Student, Recruiter, Company).
6. Users can create group.

Medium Priority

7. User can update Profile.
8. User can delete Profile.
9. Users must be able to browse jobs based on their userJobInterests and filter jobs by userRemote preference (remote work).
10. Users must be able to save or bookmark jobs (userJobBookmarks) for future reference.
11. Users must be able to upload or update their userResume.
12. Users should be able to input and edit their work history (userExperience), including the job title, company, and duration of employment.
13. Users must be able to add, update, and display their userEducation background.
14. Users must be able to add, update, and display a list of userSkills they possess.

Low Priority

15. Users should have the ability to filter job listings based on whether they are userRemote jobs or in-office.
16. Users should be able to assign userTags (keywords) to highlight their areas of expertise for recruiters or companies.
17. User can add, remove and update group as per the requirement.

18. User can request recruiters for the group

Student:

High Priority

1. Students can provide personal details such as first name, last name, email, and password during sign-up.
2. Students can upload a profile image to personalize their accounts.
3. Students enter their school/university, degree program, major, and expected graduation year.
4. Students can list skills they've acquired and specify their job interests (roles, industries).
5. Students can browse job listings and filter them based on job roles, industries, and remote preferences.
6. Students can apply to job openings by submitting their profile, resume, and any other required materials.
7. Students can update their personal, academic, and career details at any time (skills, job interests, resume, etc.).

Medium Priority

8. Students can modify or update their degree information, major, or expected graduation year as they progress.
9. Students can update their job role preferences, remote job preferences, and the industries they are interested in.
10. Students can document their internship experiences, including details about the company, role, and duration.
11. Students can create a list of academic or personal projects they've completed, including titles, descriptions, and technologies used.
12. Students can add, update, and remove skills on their profile as they gain new experiences.
13. Students can connect with and add mentors to their profile for guidance and career advice.

Low Priority

14. Students can add details about their involvement in clubs and extracurricular activities.
15. Students can delete their profile if they no longer want to use the platform, removing their data, job applications, and networking connections from the system.
16. Students can regularly update their resume to reflect their latest experiences and skills.

Companies:

High Priority

1. Companies can create a profile by providing their name, official email, website URL, industry, and physical location.
2. Companies can list new job openings by specifying job details, requirements, and responsibilities.
3. Companies can view applications submitted by students or job seekers, including resumes and cover letters.
4. Companies can describe their mission, values, and services to give potential job seekers insights into their organization.
5. Companies can upload their logo to personalize their profile and branding on the platform.
6. Companies can update any of their profile details such as the company name, location, industry, website, and description as needed.

Medium Priority

7. Companies can edit the details of their current job postings (e.g., job title, location, salary, or role responsibilities).
8. Once positions are filled, companies can close job postings or remove them from the platform.
9. Companies can interact with mentors on the platform to collaborate or provide insight into industry trends and recruitment strategies.
10. Companies can add, remove, or update tags representing their areas of expertise to ensure they remain aligned with evolving company operations.

11. Companies can advertise their brand, achievements, updates, and services on their profile, making it visible to potential job seekers and partners.

Low Priority

12. Companies can list and describe the products they offer, providing additional details for job seekers or potential clients interested in learning about their services.
13. Companies can update the details of their products if changes occur in their product lines or when they want to promote new offerings.
14. Companies can change or update their logo image if rebranding or making updates to their corporate identity.
15. Companies can delete their profile, removing all associated data including job postings, product listings, and messages from the system.

Recruiter:

High Priority

1. Recruiters can create a profile by providing their full name, official email, phone number, company name, and geographical location.
2. Recruiters can add job listings to the platform by specifying job details (title, location, responsibilities, and qualifications).
3. Recruiters can directly message candidates for further engagement, including scheduling interviews or providing feedback on applications.
4. Recruiters can update details of job openings such as the job description, salary, or role requirements.
5. Once a job opening is filled, recruiters can close or delete the job posting from the platform.

Medium Priority

6. Recruiters can define their industry focus or the specific job roles they specialize in, helping them connect with relevant job seekers.

7. Recruiters can upload a profile image to personalize their account and build trust with candidates.
8. Recruiters can modify their tags to reflect changes in their recruitment focus or industries they cover.
9. Recruiters can control the visibility of their profile, determining whether it is public or only viewable by certain groups or connections.
10. Recruiters can update their profile details such as name, contact information, and location.

Low Priority

11. Recruiters can add keywords representing their areas of focus to improve their searchability.
12. Recruiters can delete their profile, which will remove all associated data such as job postings, messages, and connections from the platform.

Group:

High Priority

1. Users can create a group by providing necessary details such as group name, description, and tags to describe the focus or purpose of the group.
2. The user who creates the group will automatically be assigned as the group creator with administrative privileges.
3. If the group is set to private, group creators or admins can approve or reject membership requests.
4. Group members can create and share posts within the group, including job openings, discussions, and announcements.
5. Members can react to posts by liking, commenting, or sharing.

Medium Priority

6. Group creators or admins can update the group name, description, and tags as the group's focus evolves.

7. Users can search for groups by name, description, tags, or industry keywords to find groups that align with their interests.
8. Users can add relevant tags or keywords to enhance the searchability of the group based on industry, roles, or interests.
9. Group members can edit or delete their own posts, and group creators or admins can remove inappropriate posts.

Low Priority

10. Users can view reactions to their posts, such as likes, comments, or shares from other members.
11. Members, especially recruiters or group admins, can share job openings within the group, including details such as job title and description

Milestone -1

Database Architecture

1. User Table

   Description: Stores personal information for all types of users (students, recruiters, companies).

Column Name	Data Type	Description
user_id	ObjectId (PK)	Unique identifier for each user.
firstName	String	First name of the user
lastName	String	Last name of the user
email	String	Email Address
Password	String	User Auth Password
userType	String	User role (Student/Recruiter/Company)
userCountry	String	Country where the user resides
userState	String	State where the user resides
userCountry	String	Country where the user resides
userCity	String	City where the user resides
userJobInterests	Array of Strings	Job categories of interest
userRemote	Boolean	Indicates if the user prefers remote jobs
userResume	String (URL)	URL/path to the user's uploaded resume
userSkills	Array of Strings	List of skills the user possesses
userExperience	Array of Objects	Work experience (title, company, years)
userEducation	Array of Objects	Educational background (degree, institution, years)
userJobBookmarks	Array of Objects	Notifications received by the user.
userTags	Array of Strings	Keywords representing user's expertise.

2. Companies Table

Description: Stores information about companies that are part of the platform.

Column_Name	Data Type	Description
company_id	ObjectId (PK)	Unique identifier for each company.
companyName	String	Name of the company.
companyEmail	String	Official email address of the company.
companyWebsite	String	URL of the company's official website.
companyIndustry	String	Industry in which the company operates.
companyLocation	String	Physical location of the company.
companySize	String	Size of the company (e.g., small, medium, large).
companyDescription	String	Description of the company's mission and services.
companyProducts	Array of Strings	List of products offered by the company.
jobOpenings	Array of ObjectId	Current job openings available (references to jobs table).
companyLogo	String (URL)	URL/path to the company logo image.
companyTags	Array of Strings	Keywords representing the company's expertise.

3. Recruiter Table

Description: Stores information about recruiters associated with companies.

Column_Name	Data Type	Description
job_id	ObjectId (PK)	Unique identifier for job
jobTitle	String	Title of the job position
jobDescription	String	Detailed description of the job
jobLocation	String	Location of the job
jobRequirements	Array of String	Required qualifications
jobSalary	String	Salary range for the job
jobType	String	Type of employment (Full/Part-time, Contract)
company_id	ObjectId (FK)	Reference to the company
recruiter_id	ObjectId (FK)	Reference to the recruiter
jobApplicationLink	String (URL)	Link to apply for the job
jobPostedDate	Date	Job posting date

4. Job Table

Description: Stores job postings made by companies or recruiters.

Column_Name	Data Type	Description
Job_id	ObjectId (PK)	Unique identifier for job posting
jobTitle	String	Title of the job position.
jobDescription	String	Job role description
jobLocation	String	Location of the job
jobRequirements	Array of Strings	Required qualifications for the job.
jobSalary	String	Salary range for the job.
jobType	String	Type of employment (Full/Part-time, Contract)
company_id	ObjectId (FK)	Reference to the company
recruiter_id	ObjectId (FK)	Reference to the recruiter
jobPostedDate	String	Date when the job was posted.
jobApplicationLink	String (URL)	Link to apply for the job (if applicable).

5. Groups Table

Description: Stores information about user-created groups.

Column_Name	Data Type	Description
group_id	ObjectId (PK)	Unique identifier for each group.
groupName	String	Name of the group.
groupDescription	String	Brief description of the group's purpose.
groupTags	Array of Strings	Tags associated with the group (e.g., industry, job roles).
members	Array of ObjectId	List of user IDs representing members of the group.
createdDate	Date	Date when the group was created.
admin_id	ObjectId (FK)	Reference to the user who created the group.

Milestone -1

Frontend-Backend Communication Architecture

The client side will be a web-based application, likely built using a JavaScript framework (React.js) which communicates with the backend via HTTP requests to the REST API endpoints. The backend server will be powered by FastAPI or a similar Python framework, with a MongoDB database for data persistence.

The general flow is as follows:

- Client makes requests to the backend REST API (e.g., to fetch job listings, user profiles, etc.).
- Server receives the request, processes it (e.g., fetches data from the database, performs business logic), and returns a JSON response back to the client.
- Client renders the data in the appropriate UI components, providing users with a seamless experience.

Key Components and Communication Flow

2.1 User Authentication and Session Management User

Actions:

- Sign Up
- Log In
- Log Out
- Password Reset

Flow: 1. Sign Up (POST

/api/v1/auth/signup):

- The client sends user data (email, password, name, etc.) to the backend.
- The backend hashes the password, creates a user in the database, and responds with a success message or error (e.g., email already taken).
- Upon success, the client may be automatically logged in, receiving a JWT (JSON Web Token) or session token for authentication.

2. Login (POST /api/v1/auth/login):

- The client sends credentials (email and password) to the backend.
- The backend verifies the credentials, and if valid, generates and returns a JWT token or sets a secure HTTP-only cookie.
- Subsequent API requests from the client must include this token (in headers) for authentication.

3. Authentication Middleware:
   - The backend will have a middleware to check the token or session in each request header and validate the user identity.

2.2 Profile Management and Updates User

Actions:

- Update Profile
- View Profile
- Update Resume/Skills Flow: 1. View Profile

(GET/api/v1/users/profile/{user_id}):

- The client requests the user profile by sending the user_id in the request URL.
- The backend retrieves the user data from the database and sends it back as a JSON response.
- The client renders the profile data on the user's dashboard or profile page.

2. Update Profile (PUT /api/v1/users/profile/{user_id}):

- The client sends an updated profile (name, location, skills, etc.) to the backend.
- The backend validates the data and updates the corresponding user record in the database.
- A confirmation response is sent back to the client.

2.3 Job Searching and Filtering User

Actions:

- Search for Jobs
- Filter Jobs by Location, Skills, Remote
- Bookmark Jobs

Flow:

1. Search Jobs (GET

/api/v1/jobs?search={query}&location={city}&remote={boolean}): o The client sends a request to search jobs based on criteria (keywords, location, remote, etc.).

- The backend queries the database for jobs that match the criteria.
- The server returns a list of job postings in a paginated format (e.g., JSON array of job objects).

2. Bookmark Job (POST /api/v1/users/bookmark):
   - The client sends the job_id that the user wants to bookmark.
   - The backend saves the bookmark in the user's bookmarked_jobs array.
   - The client receives a success response, which can trigger a visual confirmation (e.g., "Job bookmarked" notification).

2.4 Grouping and Collaboration User

Actions:

- Create a Group
- Join a Group
- Post Messages in a Group

Flow:

1. Create Group (POST /api/v1/groups/create):
   - The client sends group data (name, description, tags) to the backend.
   - The backend creates a new group entry in the groups table, assigning the user as the group admin.
   - The backend returns a confirmation message with the newly created group details.
2. Join Group (POST /api/v1/groups/join):
   - The client sends the group_id and the user_id to join the group.
   - The backend adds the user to the group's member list and sends a success message back.

3. Group Messaging (POST /api/v1/groups/{group_id}/message):
   - The client sends a message to the group.
   - The backend validates the user's membership and stores the message in the group's message history.
   - The message is then sent to all group members in real-time via WebSockets.

Database Relationships and Communication

Here's how the communication between the tables will work at the database level:

- Users ↔ Jobs: Users can bookmark jobs, apply for jobs, and track applications. This relationship is managed by foreign keys linking user_id to job_id in the jobs table.
- Users ↔ Groups: Users can join groups. Each group will have an admin (creator) and multiple members, stored in a relational join table, e.g., group_members.
- Companies ↔ Jobs: Each company posts multiple jobs, represented by a foreign key company_id in the jobstable.
- Recruiters ↔ Jobs: Recruiters manage job postings, with the recruiter_id linked to specific job postings.

Tech Stack & Tools

- Backend: FastAPI (Python), MongoDB, Redis (for caching and real-time notifications)
- Frontend: React.js (or similar), WebSockets for real-time features
- Authentication: JWT for secure user sessions
- Deployment: Docker containers, Nginx for load balancing, and AWS for cloud infrastructure

Non-Functional Requirements

Performance:

1. Response Time: The platform should respond to user requests (e.g., loading pages, fetching job recommendations) within 3 seconds on average.
2. Scalability: The platform must be able to support up to 500,000 active users simultaneously without degradation in performance.
3. Throughput: The system must handle up to 10,000 concurrent job searches and 1,000 job applications per minute during peak times.

Storage Space:

1. **Data Storage:** The platform should provide **10 TB** of total storage to handle user profiles, resumes, job postings, and group messages.
2. **Resume Storage:** Each user can upload a resume of up to **5 MB**, and the platform should handle up to **10 million resumes**.

Usability:

1. **User Interface:** The platform should have an intuitive and easy-to-navigate UI, accessible for users of all technical levels.
2. **Responsiveness:** The platform must be fully responsive and usable on desktops, tablets, and smartphones, ensuring a consistent experience.
3. **Learnability:** New users should be able to learn and navigate the platform within **15 minutes Security:**

1. **Authentication:** The platform must support multi-factor authentication (MFA) for added security.
2. **User Privacy:** Users must have control over the visibility of their profiles and data, with clear options to manage privacy settings.

Maintainability:

- **Modular Design:** The system should be modular, allowing developers to update or add new features with minimal impact on other parts of the system.
- **Error Logging:** The system must log **100% of errors** and provide detailed logs for troubleshooting and issue resolution.

Milestone -1

Competitive Analysis

Feature	Competitor-1 Handshake	Competitor-2 LinkedIn	Competitor-3 Indeed
Tailored Career Recommendations	Personalized job recommendations based on university partnerships	Limited (Job alerts based on basic profile)	Limited job suggestions
Grouping/Communities Feature	Limited to schoolbased groups	Industry groups available	No group feature
Career Skill Assessments	No dedicated assessments	No skill assessments	No assessments

The On-Board platform offers key advantages over competitors like LinkedIn, Handshake, and Indeed. It features advanced, AI-driven career recommendations tailored to users' skills and industry trends. On-Board emphasizes structured mentorship connections for direct mentor-mentee matching, which is lacking in other platforms. Additionally, it provides comprehensive job application tracking and real-time alerts to keep users organized in their job search. The grouping and communities feature fosters collaboration and networking across industries. These elements position On-Board as a user-centered solution for job seekers and professionals looking to advance their careers.

Milestone -1

High-level system requirements

1. Framework
   - **Backend:** Node.js with Express.js
   - **Frontend:** React.js (JavaScript)
   - **Database:** MongoDB (depending on data structure needs)

2. Tools
   - **Version Control:** Git
   - **Containerization:** Docker
   - **CI/CD:** GitHub Actions or Jenkins
   - **Testing:** Jest and React Testing Library for frontend; Mocha or Chai for backend

3. **Deployment Platform**
   - **Cloud Provider:** AWS (Amazon Web Services) or Azure
   - **Web Server:** Nginx or Apache for serving frontend and reverse proxy for backend
   - **Application Hosting:** AWS Elastic Beanstalk or Heroku

4. **External APIs and Libraries:**
   - **Authentication:** JSON Web Tokens (JWT) or Firebase Authentication
   - **Payment Processing:** Stripe or PayPal API (if applicable)
   - **Data Visualization:** Chart.js or D3.js
   - **File Uploads:** AWS S3 for storing user uploads (e.g., resumes, images)

Milestone -1

Team Members:

Krushna Thakkar – **Team Lead**
Chetas Parekh- **Scrum Master**
Khayal Dobaria- **Front-End Developer**
Jimmie Wu- **Back-End Dev**
Shrey Rakesh Kevadia**- Git Master**

Version	Description	Milestone
V0.1	Initial setup + Backend	1
V0.2	User login and sign-up Profile setup	
V0.3	Dark mode toggle. Job listing page	2
V0.4	Grouping Features	3
V0.5	Profile section	4
V0.6	Job search + UI	5
V0.7	Job application tracking	