

Foundations of Software Engineering: CSI23124

# Intelligent Team Formation System for University Project Environments (Team R US)

## Group Members (Sri Group 07 )

Gunarathne Janith Deshan (10676638)

Methuli Jayawickrama (10675813)

Amie Shonnelli (10647883)

Oshadi Rajapaksha (10677265)

Dineth Sandeepa Fonseka (10704505)

## Contents

1.Introduction.....	5
1.1 Purpose.....	5
1.2 Scope.....	6
1.3 Product Overview .....	7
1.3.1 Product Overview Diagram (Use Case) .....	7
1.3.4 User Roles and Expectations .....	10
1.4. System Limitations.....	11
1.5. Definitions, Acronyms, and Abbreviations .....	12
1.5.1. Computing-related Terms .....	12
1.5.2. Domain-related Terms.....	14
1.5.3 Abbreviations .....	14
2.Reference.....	15
3.Requirements .....	17
3.1 Functional Requirements of the system.....	17
3.1.1 Home screen .....	17
3.1.2 Mission Configuration .....	17
3.1.3 During a mission .....	18
3.1.4 Mission Review .....	18
3.2 Performance Requirements .....	19
3.2.1 Static Numerical Requirements .....	19
3.3 Usability Requirements .....	21
3.4 Interface Requirements .....	23
3.4.2 Hardware Interfaces.....	23
3.5 Internal Code Review Requirements.....	24
3.5.1. Code Review Standards .....	24
3.5.2 Automated Testing Requirements.....	24
4.Verification .....	24
4.1 Functionality Verification .....	25

3.1.1 Home Screen Functions (User Interface and Authentication) .....	25
3.1.2 Mission Configuration Functions (Project and Talent Pool Management) .....	26
3.1.3 During Mission Functions (Real-time Monitoring and Communication) .....	27
3.1.4. Mission Review Functions (Reporting and Feedback).....	28
4.2 Performance Requirements Verification.....	29
3.2.1 Static Numerical Requirements (System Load Tests) .....	29
3.2.2 Dynamic Numerical Requirements (Speed & Responsiveness).....	30
3.2.3 Capacity and Scalability Requirements (Growth and Expansion Testing) .....	31
3.2.4 Peak Workload Conditions (High-Demand Load Handling) .....	31
3.2.5 Data Backup and Recovery Time (System Reliability) .....	32
3.2.6 Error Handling and Response Time (Error Feedback and Recovery Tests) .....	32
4.3 Usability Testing's Verification .....	33
3.3.1. Effectiveness of the System .....	33
3.3.2 Efficiency Of the Sysem .....	34
3.3.3. Satisfaction of the user.....	35
3.3.4 Accessibility and Error Prevention .....	36
4.4 Interface Requirements Verification .....	37
3.4.1 User Interface Requirements .....	37
3.4.2 Hardware Interface Requirements.....	38
3.4.3 Software Interface Requirements.....	38
4.5 Code Review & Testing Verification .....	39
3.5.1. Code Review Standards .....	39
3.5.2 Automated Testing Requirements.....	40
5. Appendix 1 .....	41
5.1 Display Team Justification (Methuli Jayawickrama -10675813) .....	41
5.2 Update MBPT Results (Methuli Jayawickrama – 10675813).....	43
5.3 System Login (Amie Shonnelli 10647883) .....	46
5.4 Display Assigned Project (Amie Shonnelli -10647883).....	48
5.5 Upload CV (Oshadi Rajapaksha – 10677265).....	49

5.6 Receive Notifications (Oshadi Rajapaksha – 10677265) .....	52
5.8 Send Notifications (Dineth Sandeepa Fonseka - 10704505) .....	56
5.9 Generate Teams ( Janith Deshan – 10676638) .....	58
5.10 Edit Talent Pool (Janith Deshan – 10676638).....	60

# 1.Introduction

This is a document that includes details about the streamline and the process of the form of the project teams from a diverse pool of team members those who are having different skills and the experiences in different sections.

The aim of the Team R Us system is to finish a project with the support of the team members with the preferences with the project requirements. By using AI tools, it helps to reduce the false details in this project, and it will be flexible to the team members and complex in the project to achieve their task.

## 1.1 Purpose

The purpose of the Team R Us system is to simplify and enhance the formation of the project teams in a diverse group of peoples with various talents and abilities. In this project it supplies a system that resolves the tasks information of the project teams by identifying the skills and the experience with the project requirements. Administrators allow teams to focus on the activities to minimize the manual effort and the requested time, to create a process of team formation that meets all needs and preferences to increase the number of members that allows the system. To improve the educational side with the objective's requirements for the units like ECU Applied Project and the other organizations by the support of the team. A need of this system is adding AI Tools to the process for improving the performance of the tasks and it supports to refer the skills, experience of form the teams that used to the completion of project requirements.

This is helpful to make decisions by identifying the parts which team members are unable to complete, and it is useful to make better outcomes in the system. The aim of the system is to introduce solutions and face the difficulties that are affected by the organizations, hackathons and university departments. Also, the Applied Project unit at the Edith Cowan University designing well successes and creates good teams which includes from the needs of the project.

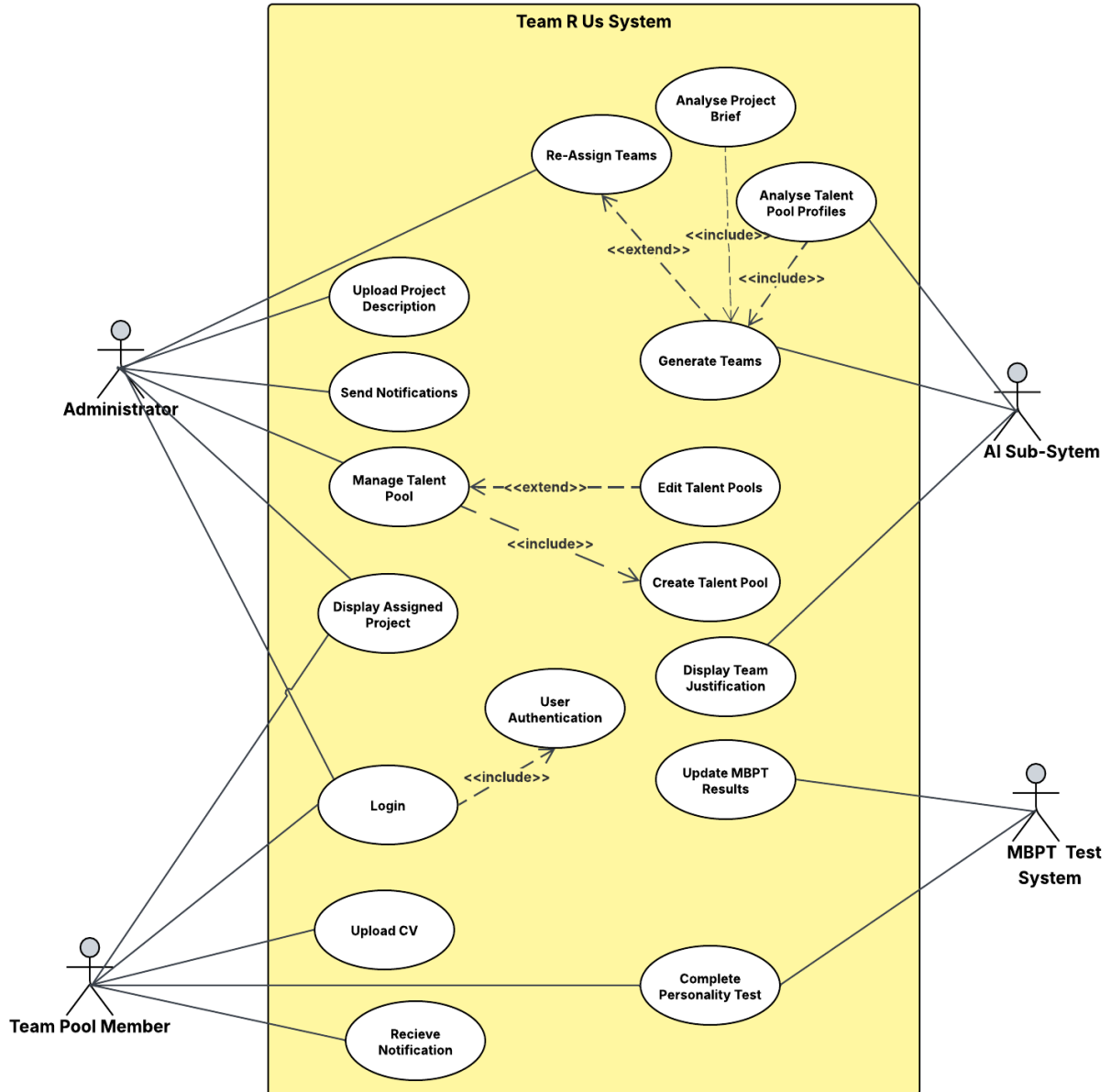
## 1.2 Scope

The System Teams R Us is certainly support in the organization with lot of information's that related to the project details and there is an ability to giving access to the administrators to get join team members by identifying their skills, background structure and personality identifications. In this system it is helpful for the team members with the support of the AI Tools to the team with formation with the messages that get useful for the administrators.

- ❖ The system use Python based desktop application designed to assist in intelligent and unbiased team formation for academic or organizational project environments.
- ❖ It provides a platform for administrators to manage project descriptions and a talent pool of users, including uploading profiles, editing details, and assigning users to projects.
- ❖ Each talent pool member profile includes
  - Personal information
  - Skills and experience
  - MBPT (Myers-Briggs ) personality test type
  - Uploaded CV in PDF format
- ❖ The system integrates with an external MBPT personality test provider with allowing members to complete the test and submit their results to the system.
- ❖ A built in AI system analyzes user profiles and project requirements to automatically generate balanced and well-suited teams.
- ❖ The AI system also gives a clear justification for each team team forms and helping administrators to understanding the logic behind assign members.
- ❖ Administrators are able to manually reassign team members if needed after AI-generated suggestions.

## 1.3 Product Overview

### 1.3.1 Product Overview Diagram (Use Case)



### 1.3.2 Functional Requirements

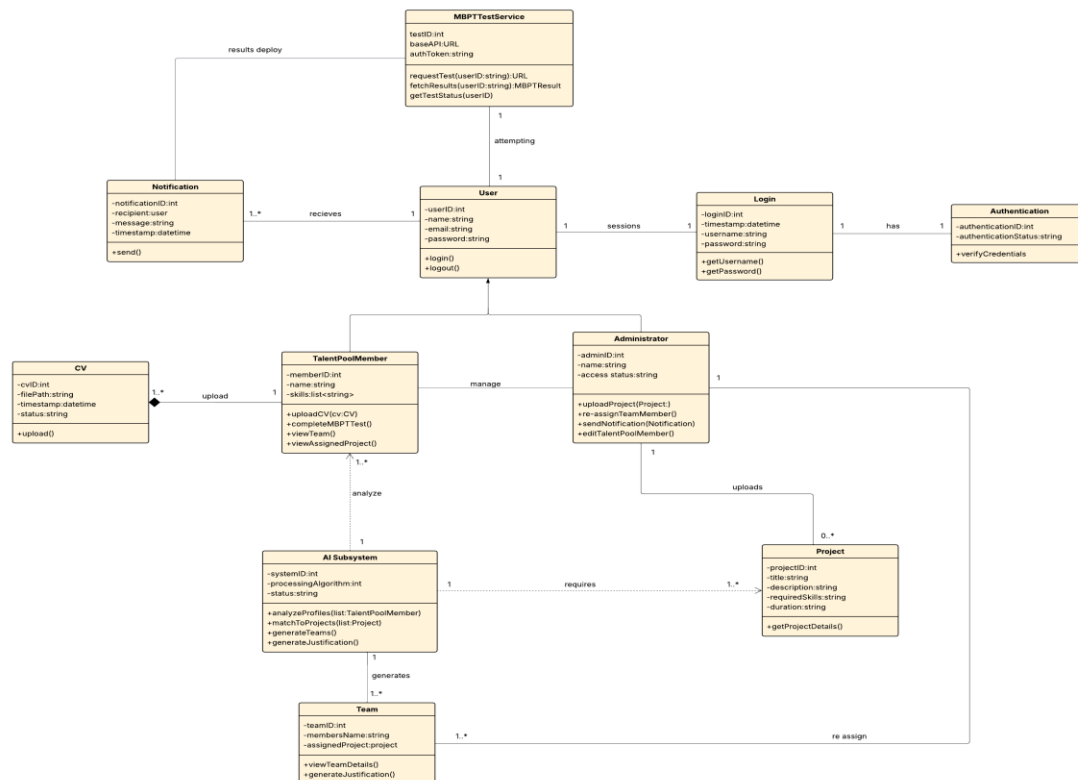
- 1.**Administrators in system can upload project descriptions in .PDF or .DOCX format, including project name, summary, required skills, and estimated duration.
- 2.**It must be possible to create and manage a Talent Pool with user details such as full name, email address, skillset, years of experience, and MBPT status.
- 3.**Admin users shall have the ability to edit Talent Pool Member information in system, including modifying skills, removing users, or updating personality test results.
- 4.**A secure login interface shall be provided, with user authentication-based Admin and pool member credentials.
- 5.**Talent Pool Members must be able to upload their CVs, limited to PDF and .DOCX format only, which are stored locally and linked to their profile for AI analysis.
- 6.**Personality testing will be integrated via third-party MBPT provider, allowing users to complete the test externally.
- 7.**MBPT results must be received from the external MBPT test system, either automatically via import or manually entered by administrators, and saved to the user's profile.
- 8.**Notifications shall be sent to the users about test completion, project assignments, or updates using the GUI notification system.
- 9.**Team generation shall be initiated by AI Sub-System, and the system will also use it to form teams by analyzing both project briefs and talent profiles (skills, MBPT type, experience).
- 10.**After team generation in the system Admins shall be able to manually reassign members.
- 11.**For each generated team, the system shall display a clear justification explaining how and why members were selected based on matching attributes.
- 12.**All users shall have access to view their with complete visibility of project title, role, team member list, and description through their profile dashboard.



### 1.3.3. Non-Functional Requirements

- 1.The system must be utilize by non-technical users and supporting workflows that require only basic computer literacy.
- 2.User interactions shall respond within 2 seconds including login, CV upload, personality test status check, and team view.
- 4.Scalability shall support up to 1000 users and 100 project entries while maintaining responsive system performance.
- 5.Role-based login and access control must be enforced bydistinguishing clearly between Administrator and Member privileges.
- 6.Integration support for the external MBPT systems through file import or manual data entry.
- 7.The system must operate on local machines (Windows 10+ or macOS), without needing internet connectivity after installation.

### Product Class Diagram



## 1.3.4 User Roles and Expectations

### 1.3.4.1 Administrator

Administrators are able to upload project briefs, manage the talent pool, initiate team formation and finally decide on team assignments. These are the following expectations from administrators

- **Educational Level** - These people are normally held to at least a tertiary performance or are pursuing higher learning activities in an institution (e.g., lecturers, tutors, or course coordinators).
- **Technical Expertise** - Intermediate to advanced skill level. Administrators are expected to be able to freely work with the web platform and perform configuration tasks such as superficial uploads or editing user personal data.
- **Experience** - Users in this category are mostly experienced in settings where teamwork is facilitated in one way or the other and appreciate an academic project workflow.
- **Disabilities Consideration** - The system assumes general accessibility, with interface elements possibly designed with a visually impaired user in mind (e.g., clear font contrast, keyboard, navigation) awaiting full implementation of accessibility standards.
- **System Usage Context** - Likely accessed through desktop computers or institutional devices, with use occurring in high-pressure activities such as preparation for assignment deadlines. The system, therefore, needs to be intuitive and fast.

### 1.3.4.2 Talent pool member

These users are individuals being assigned in teams, filling in the profiles and personality assessments that receive system-generated team notifications.

- **Educational Level** - Mostly undergraduate students but with variations in academic background. Content is to be understandable for a non-technical major.
- **Technical Expertise** - Basic to moderate. Users may be unfamiliar with technical nomenclature or multi-step workflows. Instead, the system favors guided interactions and simple interfaces.
- **Experience** - Most of the users are early in their career or educational journey and could hardly have the experience in using AI-supported platforms for team assignment.

- **Disability Consideration** - Users may be cognitively, visually, or mobility impaired. The system must reduce cognitive load and support wherever possible, such as screen readers or keyboard shortcuts.
- **System Usage Context** -These are mostly accessed by users from their personal devices-laptops and mobile phones. Hence, it requires a mobile-friendly UI and intuitive flow.

## 1.4. System Limitations

The Teams R Us system is subject to various constraints that limit development choices, platform flexibility, and runtime behavior. These constraints dictate the architectural and functional scope of the system and must be considered throughout the whole software lifecycle.

### *1.4.1.1.Regulatory Requirements and Policies*

The system must be able to comply with the Privacy Act 1988 (AU) and relevant corporate privacy policies, including the secure collection, storage, and handling of PII, CV, and MBTI personality test results, with due regard to consent and the right to know what the information is used for.

### *1.4.1.2.hardware limitations (e.g., signal timing requirements)*

There are no major limitations imposed on timing or signal processing; however, the system is designed for typical workers on web-enabled devices. In other words, devices with aging hardware and limited RAM/CPU resources may experience performance degradation, especially when used for AI-based team generation.

### *1.4.1.3.interfaces to other applications*

This system does not support native API-level integration. There is also a file-based upload facility for project briefs (e.g. DOCX/PDF) and an external redirect to a third-party MBPT personality testing platform. This hinders automatic interoperability with other organizations' systems (e.g. LMS or HR systems).

### *1.4.1.4.parallel operation*

Currently, real-time collaborative editing and multi-admin concurrent upload are not supported. Parallel operations (e.g. concurrent creation of groups in the same pool) are not properly synchronized and may cause data integrity issues.

#### *1.4.1.5.audit functions*

Some limited audit functions have been implemented. All key actions, such as team building and manual reassignments, are logged for internal tracking purposes, but detailed audit trails, including change logs and rollback capabilities for each field, are not implemented.

#### *1.4.1.6.control functions*

The system allows administrators to override AI-based allocations, but there are no fine-grained control hierarchies that facilitate multi-level organizational use, such as review approval workflows.

#### *1.4.1.7.higher-order language requirements*

The system is implemented only in English. There is no functionality or support for multilingual interfaces or localization/internationalization, which limits the ability for use outside of English without considerable customization.

#### *1.4.1.8.signal handshake (e.g., XON-XOFF, ACK-NACK)*

Not applicable. HTTP/HTTPS is used to communicate, and there are no hardware or low-level protocol interfaces.

#### *1.4.1.9.quality requirements (e.g., reliability)*

The system aims for medium reliability in academic contexts. It is not a high availability system. During peak server loads for artificial intelligence processing, or file upload, users may experience delays or temporary disfunctionality.

## 1.5. Definitions, Acronyms, and Abbreviations

### 1.5.1. Computing-related Terms

#### **1.Administrator**

An elevated user capable of managing project data, assigning teams, and overriding AI system decisions.

#### **2.AI Sub-System**

An internal intelligent module that analyzes talent profiles and makes optimal project team formations based on preset logic and compatibility metrics.

### **3.CV (Curriculum Vitae)**

The document uploaded by users giving their personal, educational, and professional detail. Accepted noted formats are PDF or DOCX.

### **4.Dashboard**

A GUI-based interface through which the user accesses the system for tasks such as file uploads, viewing team details, or viewing assignments.

### **5.GUI (Graphical User Interface)**

This is the visible part of the system built with Tkinter that allows the user to interact with it through buttons, forms, and displays.

### **6.MBPT / MBTI (Myers-Briggs Personality Type Indicator)**

An integrated psychological assessment that categorizes user personalities to assist in better team matching.

### **7.SQLite**

A local lightweight database that the system uses to store all the information about profiles, projects, and teams.

### **8.Tkinter**

A library used in Python for the creation of the desktop GUI for this system.

## 1.5.2. Domain-related Terms

### 1.Team R Us System

The proposed intelligent software solution that automates team formation for university or organizational projects on the basis of skills, experiences, and personality data.

### 2.Talent Pool Member

A user registered in the system with a profile filled in with personal data, details of skills, experience, CV, and results of a personality test. Such users are placed into project teams.

### 3.Project Description

Uploaded by administrators, this document describes the main goals of the project, skill requirements, and estimated duration; team formation is based on this document.

### 4.Team Justification Report

An explanation produced by the system that presents the rationale behind the selection of team members for a particular project.

### 5.Notification System

An internal mechanism for informing the users whenever tests are completed with assignments, important updates, and so forth, either by in-app notifications or by e-mail.

## 1.5.3 Abbreviations

Abbreviation	Definition
AI	Artificial Intelligence
CV	Curriculum Vitae
GUI	Graphical User Interface
MBPT	Myers-Briggs Personality Type
PDF	Portable Document Format

DOCX	Microsoft Word Open XML Document Format
SRS	Software Requirements Specification
JSON	JavaScript Object Notation
SQLite	Lightweight Structured Query Language database engine
WCAG	Web Content Accessibility Guidelines
API	Application Programming Interface
UAT	User Acceptance
IDE	Integrated Development Environment

## 2.Reference

1. Agile development methodologies: An essential guide | BrowserStack. (2025, January 14). *BrowserStack*. <https://www.browserstack.com/guide/agile-development-methodologies>
2. Australian Government. (1988). *Privacy Act 1988*. <https://www.legislation.gov.au/Series/C2004A03712>
3. Boehm, B. W. (1981). *Software engineering economics*. Prentice-Hall.
4. Bridges, J. (2025, April 29). How to make a project budget: Project budgeting basics (templates included). *ProjectManager*. <https://www.projectmanager.com/training/create-and-manage-project-budget>
5. Burke, R. (2013). *Project management: Planning and control techniques* (5th ed.). Burke Publishing.
6. Functional verification. (2023, June 17). *Wikipedia*. [https://en.wikipedia.org/wiki/Functional\\_verification](https://en.wikipedia.org/wiki/Functional_verification)
7. GeeksforGeeks. (2025, April 2). User acceptance testing (UAT) software testing. *GeeksforGeeks*. <https://www.geeksforgeeks.org/user-acceptance-testing-uat/>
8. Glinz, M. (2007). On non-functional requirements. *21st IEEE International Conference on Requirements Engineering*, 1–10. <https://doi.org/10.1109/RE.2007.45>
9. Hidalgo, R. J. F., & Fernandez, P. L. (2015). Functional requirements identification using item-to-item collaborative filtering. *International Journal of Information and Education Technology*, 5(10), 758–762. <https://doi.org/10.7763/ijiet.2015.v5.606>
10. Heldman, K. (2018). *Project management jumpstart* (3rd ed.). Wiley.

11. IBM. (2020). *IBM Watson Personality Insights*.  
<https://www.ibm.com/watson/services/personality-insights/>
12. Institute of Electrical and Electronics Engineers. (1990). *IEEE standard glossary of software engineering terminology (IEEE Std 610.12-1990)*. IEEE.
13. ISO. (2011). *ISO/IEC 25010:2011 Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — System and software quality models*. International Organization for Standardization.
14. ISO/IEC/IEEE. (2017). *Systems and software engineering — Software life cycle processes (ISO/IEC/IEEE 12207:2017)*. International Organization for Standardization.
15. Jama Software. (2024, September 5). Requirements verification and validation - JAMA Software. <https://www.jamasoftware.com/requirements-management-guide/requirements-validation-and-verification/requirements-verification-and-validation-for-product-teams/>
16. MindTools | Home. (n.d.). <https://www.mindtools.com/ahxvlqi/project-schedule-development>
17. Myers-Briggs Foundation. (2022). *MBTI basics*. <https://www.myersbriggs.org/my-mbti-personality-type/mbti-basics/>
18. Project Management Institute. (2017). *A guide to the project management body of knowledge (PMBOK® Guide)* (6th ed.). Project Management Institute.
19. Udoagwu, K. (2025, May 29). 6 steps for a solid code review process. *Blog Wrike*.  
<https://www.wrike.com/blog/code-review-process/>
20. Verzuh, E. (2021). *The fast forward MBA in project management* (6th ed.). Wiley.
21. Wiegers, K. E., & Beatty, J. (2017). *Software requirements* (3rd ed.). Microsoft Press.  
Wysocki, R. K. (2016). *Effective project management: Traditional, agile, extreme* (7th ed.). Wiley.



## 3.Requirements

### 3.1 Functional Requirements of the system

#### 3.1.1 Home screen

3.1.1.1. The system should display the project name and logo on the home screen.

3.1.1.2. The system should offer clearly labeled options: "New Mission", "Ongoing Missions", and "Mission Review".

3.1.1.3. The system should allow log-in by role (i.e., admin or member of the talent pool).

3.1.1.4. The system should redirect users to the appropriate dashboards after log-in depending on their role.

#### 3.1.2 Mission Configuration

3.1.2.1. The system must offer the ability to upload individual project descriptions in DOCX or PDF. The system should be able to extract pertinent information (skill required, goals) from uploaded documents.

3.1.2.2. The system must offer the ability to create a talent pool, e.g., adding, editing, and removing worker profiles.

3.1.2.3. The system must extract Myers-Briggs personality test details and save them against each worker profile.

3.1.2.4. The system should allow CVs to be uploaded and display experience and skills in profile form.

3.1.2.5. The system should form teams based on compatibility, personality, and matching skills.

3.1.2.6. The system should allow administrators to manually override automatically suggested teams.

3.1.2.7. The system should provide reasoning for team members being mapped to a specific project.

### 3.1.3 During a mission

3.1.3.1. The system should provide a real-time dashboard of team assignments and mission status.

3.1.3.2. The system should track deliverables, updates, and participation metrics.

3.1.3.3. The system should notify users of approaching deadlines and tasks that have been assigned to them.

3.1.3.4. The system should allow authorized users to send out broadcast messages or updates to team members.

3.1.3.5. The system should log all activity and updates for later review.

### 3.1.4 Mission Review

3.1.4.1. The system should have a summary report of mission outcomes, including team performance and objectives achieved.

3.1.4.2. The system should export mission data in PDF or CSV format.

3.1.4.3. The system should have the feature to allow users to provide feedback and rate their team experience.

3.1.4.4. The system should have submission of peer reviews and final assessment reports.

3.1.4.5. The system should save completed missions for future reference and quality assurance.

## 3.2 Performance Requirements

### 3.2.1 Static Numerical Requirements

3.2.1.1. The system shall support up to 150 local desktop terminals concurrently for uploading CVs, completing personality tests, and viewing project details.

3.2.1.2. The system shall support up to 70 active users to performing simultaneous team operations (uploading CVs, test data, or viewing team assignments). During at the peak usage the system will support up to 100 concurrent users without performance degradation.

3.2.1.3. The system shall manage data related to Talent Pool Members, including names, email, MBTI results, expertise areas, and CV uploads. Also the system shall store information for up to 100 projects and 2000 students profiles, with multiple data attributes.

### 3.2.2 Dynamic Numerical Requirements

3.2.2.1. 95% of team generation or profile update transactions shall be processed in under 2 seconds and all transactions must complete in less than 5 seconds.

**3.2.2.2.** The system shall upload and save CVs or project descriptions in under 3 seconds for 95% of cases. Maximum allowed upload processing time is 5 seconds.

**3.2.2.3.** The AI team generation system shall create optimized teams in under 5 seconds and explanations for team formations shall be displayed within 1 second of request.

**3.2.2.4.** Notifications (email or GUI prompts) to users regarding personality tests and team updates shall be issued within 30 seconds of the trigger event and 90% of notifications shall appear in-app in under 5 seconds.

### **3.2.3 Capacity and Scalability Requirements**

**3.2.3.1.** The system shall maintain data for 1000 talent pool users and 100 projects with no slowdown and system will scale to support an additional 25% users during critical updates or semester deadlines.

### **3.2.4 Peak Workload Conditions**

**3.2.4.1.** The system shall process up to 100 team assignments within a 1-hour period without delay. It shall handle 500 notifications or test result imports in a 24-hour cycle during peak loads.

### **3.2.5 Data Backup and Recovery Time**

**3.2.5.1.** Local system data shall be backed up automatically every 7 days and in case of a crash or data loss, system recovery shall occur within 15 minutes using the most recent backup.

### 3.2.6 Error Handling and Response Time

**3.2.6.1.** Any system errors (e.g., failed login, invalid file format) shall be reported in under 2 seconds also, if communication with any external system (e.g., MBTI test system) fails, a retry shall occur with a 5-second timeout and visible error feedback.

## 3.3 Usability Requirements

### 3.3.1 Effectiveness of the System

**3.3.1.1.** At least 95% of users (administrators and talent pool members) shall complete essential tasks (e.g., registering a profile, generating teams, uploading CVs) successfully without external assistance.

**3.3.1.2.** The system shall allow users to complete core tasks with an error rate of less than 5%.

**3.3.1.3.** Clear and immediate error messages shall be displayed whenever a user performs an invalid action.

**3.3.1.4.** All displayed data (e.g., team members, personality test results, uploaded documents) shall reflect the most recent information input by the user in real-time.

### 3.3.2 Efficiency of the System

**3.3.2.1.** Users shall be able to upload a CV, complete a personality profile, and view their assigned team within 5 minutes.

**3.3.2.2.** Administrators shall be able to create and assign a team to a project within 3 minutes using auto-generation.

**3.3.2.3.** Key tasks (upload CV, view teams, assign MBTI) shall be accessible within 3 clicks from the main menu.

3.3.2.4. The graphical user interface (GUI) shall provide breadcrumb-style navigation or backtracking buttons to facilitate easy navigation.

### 3.3.3 Satisfaction of Users

3.3.3.1. At least 80% of users shall rate the system as “satisfied” or better in post-use surveys (based on a 5-point scale).

3.3.3.2. At least 90% of users shall be able to complete their primary tasks without consulting a manual.

3.3.3.3. The system shall display form hints and guidance to assist users in entering correct data and reduce errors.

### 3.3.4 Accessibility and Error Prevention

3.3.4.1. Invalid inputs (e.g., unsupported file formats, empty required fields) shall be highlighted with real-time alerts.

3.3.4.2. Suggestions for correcting errors shall be provided alongside alerts (e.g., “Only PDF files are accepted”).

3.3.4.3. The system shall support keyboard navigation for all core actions to enhance accessibility.

3.3.4.4. The system shall provide a high contrast mode compliant with WCAG 2.1 Level AA standards for visually impaired users.

## 3.4 Interface Requirements

### 3.4.1 User Interface

3.4.1.1. The system shall use a desktop-based graphical user interface (GUI) built with Tkinter (Python).

3.4.1.2. All forms, tables, and buttons shall be clearly labeled and aligned logically.

3.4.1.3. A consistent color scheme and font shall be used throughout the system.

3.4.1.4. System popups (success, error, confirmation) shall follow consistent positioning and styling.

3.4.1.5. The GUI shall adapt correctly to a minimum screen size of 1366x768 pixels.

### 3.4.2 Hardware Interfaces

3.4.2.1. The system shall be installed and run on standard desktops/laptops with no additional hardware dependencies.

3.4.2.1.2. A mouse and keyboard shall be the primary input devices.

### 3.4.2 Software Interfaces

3.4.3.1. The system shall communicate with a local SQLite database or file-based JSON structure for storing data.

3.4.3.2. The system shall support manual integration with external MBTI test results (via file upload or entry form).

3.4.3.3. All files uploaded (CVs, project descriptions) must be in .PDF or .DOCX formats.

3.4.3.4. The system shall support exporting team assignments in CSV or PDF format.

## 3.5 Internal Code Review Requirements

### 3.5.1. Code Review Standards

3.5.1.1 All source code must undergo a peer review by a minimum of two qualified team members before being merged into the master codebase. Mandatory reviews apply to all major modules, including GUI components (Tkinter) Database access logic (SQLite queries) AI integration algorithms User authentication mechanisms File upload handlers Each review session must last no less than 30 minutes, and must include written documentation of findings and signed approval from reviewers before integration.

3.5.1.2 Code should follow consistent naming rules: variables in camelCase and classes in PascalCase. Every function must be documented with clear docstrings that explain the purpose, inputs, return values, and exceptions. Additionally, complex logic or algorithms should include inline comments to improve readability and maintainability.

3.5.1.3 Critical functions, such as the team generation logic, MBTI integration, and security mechanisms, must be closely evaluated for correctness, efficiency, and fairness. The AI module should be specifically reviewed to ensure balanced team allocation, prevent bias, and support diversity in team formation

### 3.5.2 Automated Testing Requirements

3.5.2.1 All functional units of the system—such as login, file upload, CV validation, team formation logic, and personality result parsing—must be covered by unit tests. These tests should verify expected outputs for valid inputs, handle edge cases, and confirm correct exception handling. Python's unittest or pytest framework is recommended.

## 4. Verification

This section illustrates how functional requirements explained in Section 3.1 will be verified through testing and measurement.



## 4.1 Functionality Verification

### 3.1.1 Home Screen Functions (User Interface and Authentication)

This section verifies the fundamental user interface functionality and role-based access control.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
HSF01	Verify that project name and logo display correctly on home screen.	Visual inspection and automated UI testing to ensure presence and correct positioning of the logo and project name.	3.1.1.1 Project name and logo display
HSF02	Test that all three navigation options ("New Mission", "Ongoing Missions", "Mission Review") are correctly labeled and work correctly.	Manual and automated testing of button label, accessibility, and navigation feature.	3.1.1.2 Navigation options display
HSF03	Verify role-based login functionality for admin role and talent pool member role.	Testing login process with different user credentials and ensuring role allocation.	3.1.1.3 Role-based login
HSF04	Test redirection to dashboard depending upon the user role after successful login.	Login with different roles and ensure correct dashboard displayed for every type of role.	3.1.1.4 Role-based dashboard redirection

### 3.1.2 Mission Configuration Functions (Project and Talent Pool Management)

These tests ensure the preliminary configuration features like file upload, data pulling, and team creation.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
MCF01	Test the upload of project description in DOCX and PDF format and verify information extraction.	Upload sample project files and verify accurate parsing of goals, requirements, and skills.	3.1.2.1 Project description upload and extraction
MCF02	Test editing and deletion features of talent pools.	Create, modify, and delete worker profiles and verify data persistence and integrity.	3.1.2.2 Talent pool management
MCF03	Test data extraction and profile association of Myers-Briggs personality tests.	Import MBTI test results and verify accurate association with worker profiles.	3.1.2.3 MBTI data extraction and storage
MCF04	Test CV upload functionality and the display of skills/experience.	Upload various CV file formats and verify accurate parsing and profile presentation.	3.1.2.4 CV upload and profile display
MCF05	Test automated team creation on the basis of compatibility, personality, and skills.	Form teams from various talent pools and projects and verify team formation logic.	3.1.2.5 Automated team formation
MCF06	Test administrator override option for team assignment.	Manually modify auto-created teams and verify	3.1.2.6 Manual team override

Test Case ID	Test Case Description	Verification Method	Requirement Reference
		changes saved and displayed.	
MCF07	Test team creation reasoning and justification display.	Form teams and provide team member assignments with detailed descriptions.	3.1.2.7 Team formation justification

### 3.1.3 During Mission Functions (Real-time Monitoring and Communication)

This section verifies real-time dashboard functionality, tracking capabilities, and communication features.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
DMF01	Test the display of team assignments and mission status in real-time dashboard.	Monitor dashboard updates during course of simulated mission activities and validate real-time data mirroring.	3.1.3.1 Real-time dashboard
DMF02	Provide tracking of deliverables, updates, and participation measures.	Create and refresh deliverables, validate tracking accuracy and calculation of metrics.	3.1.3.2 Progress tracking
DMF03	Test notification system for impending deadlines and task assignment.	Create deadlines and tasks, validate notifications are being sent to target users.	3.1.3.3 Deadline and task notifications
DMF04	Provide broadcast messaging functionality to approved users.	Send broadcast messages using admin user accounts and validate delivery to team members.	3.1.3.4 Broadcast messaging

Test Case ID	Test Case Description	Verification Method	Requirement Reference
DMF05	Test activity logging and audit trail feature.	Perform various system activities and validate complete logging for review.	3.1.3.5 Activity logging

### 3.1.4. Mission Review Functions (Reporting and Feedback)

This section validates reporting functionality, data export capabilities, and feedback collection functions.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
MRF01	Verify generation of summary report for mission outcome and team performance.	Conduct simulated missions and generate reports, verify accuracy and completeness.	3.1.4.1 Mission outcome reports
MRF02	Test data export functionality in PDF and CSV formats.	Export mission data both formats and verify integrity of files and accuracy of data.	3.1.4.2 Data export functionality
MRF03	Verify user feedback and team experience rating functionalities.	Submit ratings and feedback, verify data storage and collection.	3.1.4.3 Feedback and rating system
MRF04	Test peer review and final assessment report submission.	Offer peer review and assessment, proper collection and processing.	3.1.4.4 Peer review submission
MRF05	Verify mission archival and reference functionalities.	Complete assignments and proper archiving to allow	3.1.4.5 Mission archival system

Test Case ID	Test Case Description	Verification Method	Requirement Reference
		future usage and quality control.	

## 4.2 Performance Requirements Verification

This section illustrates the how performance requirements explain at Section 3.2 will be verified through testing and measurement.

### 3.2.1 Static Numerical Requirements (System Load Tests)

The static numerical requirements verifies how the system handles expected user data amount, project count, and device connections.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
SNR01	Simulate 100 desktop terminals utilizing features like registration, CV upload, and team viewing	The virtual instances that simulate desktop terminals used for load testing.	3.2.1.1 Amount of terminals to be supported
SNR02	During peak hours, simulate 100 users doing tasks like uploading data or getting alerts.	Automated scripts are used to simulate 100 concurrent users in order to stress test the system.	3.2.1.2 Number of simultaneous users to be supported
SNR03	To evaluate the system's ability to handle data, load fake data for 100	To verify the storage and retrieval of huge amounts of data, use a database to	3.2.1.3 Type of information to be handled

	projects and 1000 workers.	workers population test.	
--	----------------------------	--------------------------	--

### 3.2.2 Dynamic Numerical Requirements (Speed & Responsiveness)

This section tests the system's ability to perform tasks like generating teams, sending notifications, and processing files within the required time.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
DNR01	Measure the average transaction speed by simulating common user tasks.	Set timers to record when team creation, uploads, and profile submissions are finished.	3.2.2.1 Transaction Processing Speed
DNR02	Test file upload times for CVs and project descriptions.	Upload 50 documents and set time the operation duration	3.2.2.2 File Upload Speed
DNR03	Generates 50 teams and track how long it takes them to do tasks.	Set time the creation of each team and document its results.	3.2.2.3 Team Formation and Justification
	Send simulated notifications to 100 users and measure its delivery speed.	Use logging and GUI alerts to verify the display of notifications.	3.2.2.4 Notification System Performance

### 3.2.3 Capacity and Scalability Requirements (Growth and Expansion Testing)

This section evaluates the system's capacity ability to scale under increasing workloads and maintain performance.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
CSR01	Verify access performance by inserting fake information for 1000 profiles	Testing of retrieval processes by manual and automated under heavy data loads.	3.2.3 Capacity
CSR02	Evaluate whether the system can manage a +25% load by simulating user growth.	Run load tests using a baseline user count of 125% and track the results.	3.2.4 Scalability

### 3.2.4 Peak Workload Conditions (High-Demand Load Handling)

This section ensures the system can operate smoothly during peak hours such as assignment deadlines or bulk test result imports.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
PWC01	Simulate assignments in 100 teams for one hour.	Testing of retrieval processes by manual and automated under heavy data loads.	3.2.4 Peak Registration Loads
PWC02	Simulate 500 personality test results imports and	To replicate batch imports and notification	3.2.4 Peak Data Volume Handling

	500 notifications in one day.	dispatching to use a scheduled script.	
--	-------------------------------	----------------------------------------	--

### 3.2.5 Data Backup and Recovery Time (System Reliability)

This evaluates system's ability to recover from failures and protect user data.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
DBRT01	Run a system failure simulation and start the recovery from the most recent backup data.	Manual shutdown and restoration using a copy of backup and timed for performance.	3.2.5 Data Backup and Recovery Time

### 3.2.6 Error Handling and Response Time (Error Feedback and Recovery Tests)

The system provides real-time feedback to users when errors occur and automatically attempts to recover when needed.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
EHRT01	Measure response time by simulating file format, login or personality test failures.	Generate validation errors and verify that messages are appear within one second by use logging.	3.2.6 Error Notifications
PWC02	Simulate communication	Every five seconds, observe a retry attempt while	3.2.6 Communication Errors



	failure with external testing system.	simulating a delayed response or a faulted API.	
--	---------------------------------------	-------------------------------------------------	--

## 4.3 Usability Testing's Verification

### 3.3.1. Effectiveness of the System

Users Profile Registrations and team generation rate verify by testing its success rate of each combination in the system

Test Case Id	Test Case Description	Verification Method	Requirement Reference
EFFE 01	Verify the profile registration with a success rate without any external assistance	Combine with some users and register a profile and test and measure the success rate	3.3.1.1 Profile Registration
EFFE 02	Verify the success rate of the upload cv option	Ensure that the when uploading a cv it has completed the success rate	3.3.1.1 Upload CV
EFFE 03	Verify the generation of the team success rate	Generate a success team with a good success rate	3.3.1.1 Team Generation
EFFE 04	Measure the error rates during the tasks	The errors must be low, the calculated error must be < 5% per task.	3.3.1.2 Low error rate

### 3.3.2 Efficiency Of the Sysem

Test the efficiency of system has capable for upload the supportive documents and time flexibility.

Test Case Id	Test Case Description	Verification Method	Requirements
EFFI 01	Verify that the user completed the upload of the cv and the personality of the profile with the view of the team within 5 minutes	The users perform all tasks correctly during the given time	3.3.2.1 Task Completion Time User
EFFI 02	Verification of the admins that create and assigns a team to a project	Ensure that the admin will perform the creation of the team by the use of the auto generation	3.3.2.1 Task Completion Time Admin
EFFI 03	Verify that the cv uploaded within three clicks from the main menu	The user can enter for the cv uploaded interface by the system page	3.3.2.2 Quick Access Tasks
EFFI 4	Verification of the view teams within three clicks from the main menu	The user can log into the view team page from the system	3.3.2.2 Navigation Support

### 3.3.3. Satisfaction of the user

Checks the users satisfaction of the system by grouping various people and simulate a test in the system.

Test Case Id	Test Case Description	Verification Method	Requirements
SAT 01	Verify that 80% of users in rate system are satisfied in the post use surveys	Confirm that 80% of users rate the system as satisfied or better in post use surveys	3.3.3.1 User Satisfaction
SAT 02	Verification of the users that complete the primary task without manual assistance	Assigned that the users have completed their tasks without any support of the any assistance with a better success rate	3.3.3.2 Manual Free Task Completion
SAT 03	Verify the user entered the correct data it will display from hints	Check the visibility of the hint for each data that entering to the system	3.3.3.2 Form Hints

### 3.3.4 Accessibility and Error Prevention

Utilizing the error handling methods to able to user identify from their front end as well as system faults and bugs from the back end in the system

Test Case Id	Test Case Description	Verification Method	Requirements
ACEP 1	Verify whether the real time highlights are valid for the invalid inputs or not	Ensure the validation of the field and display that the field is required and confirm that the fields are highlighted	3.3.4.1 Invalid Input Alert
ACP 2	Verify suggestions for the unsupported file formats	Confirm that the uploaded file is in valid format and it display the format is supported to the system	3.3.4.1 Error Correction Hints
ACP 3	Verify the keyboard navigations for the all core actions in the system	If the core actions are supported with the WACG 2.1 it is complete in validation	3.3.4.2 Keyboard navigation
ACP 4	Validation with the high contrast mode for users in compliance with WCAG 2.1	It surely enables high contrast mode with the all matching options and match with the WCAG 2.1	3.3.4.2 High Contrast Mode

## 4.4 Interface Requirements Verification

This section details how structured testing will be used to validate interface-related criteria from Section 3.4. requirements.

### 3.4.1 User Interface Requirements

This section checks the system's visual and interactive features to make sure the graphical user interface (GUI) is responsive, consistent, easy to use, and compatible with standard screens.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
UIR01	Verify system uses a Tkinter based GUI	Inspect source code for Tkinter imports and usage	3.4.1.1 GUI with Tkinter
UIR02	Ensure all forms, tables, and buttons are logically named and aligned.	GUI walkthrough, checklist-based UI inspection	3.4.1.2 Labeled and aligned forms/buttons
UIR03	Ensure a uniform font and color design across all screens.	Visual inspection and UI style guide comparison	3.4.1.3 Consistent color and font
UIR04	Verify that all of the popups have a consistent appearance and design.	Activate pop-ups and compare styling and location.	3.4.1.4 Consistent popup styling
UIR05	Verify that the GUI appropriately adapts to a 1366x768 screen resolution.	Run the application in a screen emulator or resize test.	3.4.1.5 Minimum screen resolution support

### 3.4.2 Hardware Interface Requirements

This section verifies that the system works properly with common desktop and laptop hardware, especially that it is compatible with keyboard and mouse inputs and doesn't require any other devices.

Test case ID	Test case Description	Verification Method	Requirement Reference
HW01	Confirm that the system runs on regular desktops/laptops without any additional hardware.	Install and run system on basic test machine	3.4.2.1  No extra hardware required
HW02	Check sure the input is functional using a keyboard and mouse.	During system testing, only use a keyboard and mouse.	3.4.1.2  Mouse and keyboard input

### 3.4.3 Software Interface Requirements

This section authorizes the system's integration with external files and data storage, including SQLite/JSON compatibility, file format support, and successful data export functionalities.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
SW01	Confirm that the system uses SQLite or JSON for local data storage.	Inspect the configuration or codebase for data handling.	3.4.3.1  SQLite or JSON data storage
SW02	Verify that MBTI test data can be manually integrated.	Upload sample MBTI files or enter data via the form.	3.4.3.2  Manual MBTI integration

SW03	Ensure that uploaded files are only accepted in.pdf or.docx format.	Attempt uploads in numerous formats and ensure file validity.	3.4.3.3 Only PDF/DOCX uploads allowed
SW04	Ensure that the system supports exporting team data in CSV or PDF format.	Trigger the export function and verify the file content and format.	3.4.3.4 Export assignments as CSV or PDF

## 4.5 Code Review & Testing Verification

### 3.5.1. Code Review Standards

Testing the significance of source code materials that use to handling data and checking the formats of database and uploads.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
SW01	Confirm that the system uses SQLite or JSON for local data storage.	Inspect the configuration or codebase for data handling.	3.5.1.1 SQLite or JSON data storage
SW02	Verify that MBTI test data can be manually integrated.	Upload sample MBTI files or enter data via the form.	3.5.1.2 Manual MBTI integration
SW03	Ensure that uploaded files are	Attempt uploads in numerous formats	3.5.1.2 Only PDF/DOCX uploads allowed

	only accepted in.pdf or.docx format.	and ensure file validity.	
--	--------------------------------------	---------------------------	--

### 3.5.2 Automated Testing Requirements

To check the results from personality tests and other major parts of the systems testing those by manually integrating.

Test Case ID	Test Case Description	Verification Method	Requirement Reference
SW01	Ensure that the system supports exporting team data in CSV or PDF format.	Trigger the export function and verify the file content and format.	3.5.2.1 Export assignments as CSV or PDF
SW02	Verify that MBTI test data can be manually integrated.	Upload sample MBTI files or enter data via the form.	3.5.2.2 Manual MBTI integration



## 5. Appendix 1

### 5.1 Display Team Justification (Methuli Jayawickrama -10675813)

USE CASE Title	Display Team Justification - a contract for the behavior of the System in question
Primary Actor	Administrator - the stakeholder who initiates an interaction with the Sig to achieve a goal
Scope	Team R Us System - identifies the Sig
Level	User Goal - Primary Task (a single sitting)
Preconditions	That which we already have is the state of the world - specifies what must be true before the use case runs: Administrator is logged in, teams must have been created, and team formation algorithms must have completed processing
Minimal Guarantees	State of the world if undogged - states what needs to be true after the use case runs and finishes in the alternate state: System has data integrity, user session is still valid, and team data is not updated
Success Guarantees	World state on successful termination - specifies what ought to happen once the use case is run: Team justification data is displayed correctly with detailed rationale, criteria used, and member compatibility report

Trigger		Action on the system that triggers the use case: Administrator selects a specific team and requests viewing the team justification information
Main Success Scenario	Step	Action
	1	The administrator opens the team management screen interface
	2	The system presents a list of generated teams
	3	One of the teams is selected by the administrator for justification review
	4	The system retrieves the stored data on team formation criteria and team member data
	5	The System performs analysis on MBTI compatibility scores with members and their respective skills
	6	The system generates the justification report, which includes member and team balance strengths
	7	System displays a full team justification, including some visual representations
	8	The administrator goes through the justification details and reasoning
Extensions	Step	Branching Action
	4a	The available team data are incomplete: the system issues a warning and only shows whatever data is available
	5a	MBTI data missing for some members: System indicates incomplete analysis and suggests completing the data

	6a	Justification generation fails: the system issues an error message and logs the problem to be reviewed by technical team
	7a	Display formatting error: display the data in an alternative text format
Sub Variations		Branching Actions
	3a	Multiple team selection: The System allows a comparison view regarding multiple team justifications
	7a	Export Justification: Administrator can export the justification report into PDF or document format
	8a	Detailed member analysis: Administrators can drill down into member contributions

## 5.2 Update MBPT Results (Methuli Jayawickrama – 10675813)

USE CASE Title	Update MBTI Results - a contract for the behavior of the System in question
Primary Actor	MBTI 3rd Party Test System - the external stakeholder that provides MBTI assessment results

Scope		Team R Us System - identifies the Sig
Level		User Goal - Primary Task (a single sitting)
Preconditions		We expect what is already the case with the world - describes what must be the case prior to the execution of the use case: The User will have completed the MBTI personality test, the 3rd party system will have processed the results through, and the communication channel for the systems is established
Minimal Guarantees		World description, if detached, is what must retain after the use case runs to completion and finishes in the other state: Original MBTI data undisturbed, system logs tried transaction, user profile consistency.
Success Guarantees		World state upon successful termination - what must hold on completion of running use case: MBTI results are successfully updated to user profile, team compatibility algorithms are called to recalculate, and user is notified of profile update
Trigger		Action on system that triggers the use case: MBTI 3rd Party Test System notifies Team R Us System with completed test results
Main Success Scenario	Step	Action
	1	MBTI 3rd Party Test System sends results via API

	2	Team R Us System accepts and processes incoming MBTI data format
	3	System checks source and identifies user
	4	System parses MBTI results (personality type, dimension scores, confidence levels)
	5	System updates user's talent pool profile with new MBTI results
	6	System triggers team compatibility recalculation algorithms
	7	System logs successful update and creates audit trail
	8	System notifies back to MBTI 3rd Party Test System
	9	System notifies user of completion of profile update
Extensions	Step	Branching Action
	2a	Invalid format of data received: System declines update and asks for resubmission of data
	3a	Incomplete MBTI results: System requests 3rd party system for full assessment data
	4a	Incomplete MBTI results: System requests 3rd party system for full assessment data
	5a	Failure to update database: System undoes changes and notifies technical support  Branching Actions

Sub Variations		Branching Actions
	1a	Batch update processing: The System updates multiple MBTI results in batch
	5a	Comparison of profiles: The System contrasts new results against historical MBTI records and reports changes
	9a	Enhanced notification: The System provides a concise description of the manner in which the updated MBTI affects team eligibility

### 5.3 System Login (Amie Shonnelli 10647883 )

Use Case Title	Login
Primary Actor	Administrator or Talent Pool Member
Scope	Team R Us System
Level	Primary Task

Preconditions		<p>The user has to be already registered in the system.</p> <p>The system has to be connected to the authentication service.</p>
Minimal Guarantees		The user is presented an error message stating that the login attempt failed.
Success Guarantees		The user is authorized and redirected to the proper dashboard (admin or talent pool member).
Trigger		User clicks the “Login” button on the homepage.
Main success scenario	Step	Action
	1	User clicks on the “Login” button.
	2	System displays the username and password fields.
	3	User enters their credentials and clicks “Submit.”
	4	System verifies the provided credentials with the authentication service.
	5	System logs in the user and displays their relevant dashboard.
Extensions	Step	Branching Action
	3a	User leaves fields empty → system displays validation error message.
	4a	Invalid credentials → system displays “Invalid username or password.”

	5a	Login successful but role not assigned → system displays “Access denied.”
--	----	---------------------------------------------------------------------------

#### 5.4 Display Assigned Project (Amie Shonnelli -10647883)

Use Case Title		Display Assigned Project
Primary Actor		Talent Pool Member
Scope		Team R Us System
Level		Primary Task
Preconditions		<p>The user has to be already logged in to the system.</p> <p>The team generation process has to be completed by the administrator.</p> <p>The user has to be assigned to a team and a project.</p>
Minimal Guarantees		<p>If no project has been assigned yet, the system displays an appropriate message.</p> <p>The user remains on the dashboard without any errors.</p>
Success Guarantees		The system displays the relevant project title, description, and related team information to the assigned user.
Trigger		The user selects the “My Projects” option from the dashboard.
Main success scenario	Step	Action



	1	User logs into the system and navigates through the dashboard.
	2	User clicks on the “My Projects” option.
	3	System retrieves the project assignment data from the database.
	4	System displays the project title, summary, team members, and deadlines.
	5	User can view the full project brief and team information.
Extensions	Step	Branching Action
	3a	No project assigned yet → system displays: “You have not been assigned a project yet.”
	3b	Database error or delay → system displays: “Unable to retrieve project details at this time.”
	4a	Some project details are missing → system displays what is available and notes incomplete data.

### 5.5 Upload CV (Oshadi Rajapaksha – 10677265)

USE CASE Title	Upload CV - a contract for a behavior of the System in question (Sig)
Primary Actor	Talent Pool Member - the stakeholder who initiates an interaction with the Sig to achieve a goal

Scope			Team R Us System - identifies the Sig
Level			User Goal - Summary (describe a multi-sitting), Primary Task (a single sitting), Subfunction (a part of a user goal)
Preconditions			What we expect is already the state of the world - describes what must be true before the use case runs: User must be registered as a Talent Pool Member and authenticated in the system.
Minimal Guarantees			The state of the world if undogged - describes what must be true after the use case runs and finishes at the alternative state: System maintains data integrity, user session remains active.
Success Guarantees			The state of the world upon successful completion - describes what must be true after the use case runs: CV is successfully uploaded, stored in the system database, and associated with the user's profile.
Trigger			The action upon the system that starts the use case: Talent Pool Member clicks "Upload CV" button or navigates to CV upload section.
Main Success Scenario	Step	Action	
	1	System displays CV upload interface with file selection option	

	2	Talent Pool Member selects CV file from their device (PDF, DOC, or DOCX format)
	3	System validates file format and size constraints
	4	Talent Pool Member confirms upload action
	5	System processes and stores the CV in the database
	6	System updates user profile with CV information
	7	System displays success confirmation message
Extensions	Step	Branching Action
	3a	Invalid file format detected: System displays error message and prompts user to select valid format
	3b	File size exceeds limit: System displays size limit error and requests smaller file
	5a	Upload fails due to network issues: System displays retry option and saves progress
	5b	Database storage error: System logs error and displays technical support contact information
Sub Variations		Branching Actions
	2a	User uploads replacement CV: System prompts to confirm overwriting existing CV

	2b	User uploads multiple CV versions: System allows selection of primary CV
--	----	--------------------------------------------------------------------------

## 5.6 Receive Notifications (Oshadi Rajapaksha – 10677265)

USE CASE Title	Receive Notifications - a contract for a behavior of the System in question (Sig)
Primary Actor	Talent Pool Member - the stakeholder who receives notifications from the system
Scope	Team R Us System - identifies the Sig
Level	User Goal - Summary (describe a multi-sitting), Primary Task (a single sitting), Subfunction (a part of a user goal)
Preconditions	What we expect is already the state of the world - describes what must be true before the use case runs: User must be registered, have valid contact information, and notification preferences set
Minimal Guarantees	The state of the world if undogged - describes what must be true after the use case runs and finishes at the alternative state: System maintains notification queue and user preferences remain unchanged
Success Guarantees	The state of the world upon successful completion - describes what must be true after the use case runs: Notification is successfully delivered to user through preferred communication channel and marked as delivered in system

Trigger		The action upon the system that starts the use case: System-generated event occurs (team assignment, project update, or administrative announcement)
Main Success Scenario	Step	Action
	1	System detects trigger event requiring user notification
	2	System retrieves user's notification preferences and contact information
	3	System formats notification message based on event type and user preferences
	4	System sends notification through preferred channel (email, SMS, or in-app)
	5	System receives delivery confirmation
	6	System updates notification status as "delivered"
	7	User receives and views notification
Extensions	Step	Branching Action
	4a	Email delivery fails: System attempts alternative contact method or queues for retry
	4b	User preferences not set: System uses default notification settings

	5a	Delivery confirmation not received: System marks as "pending" and schedules retry
	7a	User marks notification as read: System updates status to "read"
Sub Variations		Branching Actions
	1a	Bulk notifications: System processes multiple recipients with batching mechanism
	1b	Priority notification: System bypasses normal queue and sends immediately
	3a	Personalized content: System customizes message based on user profile and history

### 5.7. Upload Project Description (Dineth Sandeepa Fonseka - 10704505)

Use Case Title	Upload Project Description
Primary Actor	Administrator
Scope	Team R Us System
Level	Primary Task (A Single Sitting)

Preconditions		<p>Admin must already log into the system</p> <p>The project file has to be ready</p>
Minimal Guarantees		If some project data is failing the system ensure that the project data is not be remove from the system.
Success Guarantees		The uploaded project description is saved into the system.
Trigger		Admin click the button Upload project description.
Main success scenario	Step	Action
	1	The admin access to the system and load the project section.
	2	The upload interface display in the system.
	3	Administrator select the upload option and upload the file.
	4	The system checks and confirm the validation of the file format.
	5	System confirm that the file was uploaded successfully.
Extensions	Step	Branching Action

	4a	Invalid File format – When an administrator uploaded a file with errors it displays an error.
Sub Variations		Branching Actions
	4a.1	Unsupported format – If the administrator uploads a file with an invalid format the system will remind to convert the file to the current version.

## 5.8 Send Notifications (Dineth Sandeepa Fonseka - 10704505)

Use Case Title	Send notifications to the users
Primary Actor	Administrator
Scope	Team R Us System
Level	Primary Task
Preconditions	Administrator must log into the system.  The administrator must inform the users about the any updates or remainders as a drafted message.
Minimal Guarantees	The notification has been saved as draft in the system
Success Guarantees	When the administrator sends the notification, it is successfully delivered to the users.
Trigger	Administrator select the option



Main success scenario	Step	Action
	1	Administrator select the users to send the notifications.
	2	After selecting the users, the administrator enters the message to send.
	3	When the administrator gives the process to the system to send the message the system sends the message to the users.
Extensions	Step	Branching Actions
	3a	When the administrator sends the notifications if there is an error it will display a retry option to the administrator to resend the notification again
Sub Variations		Branching Actions
	3a.1	If there any update or reminders for the users, any imporation information to the users the system will process notifications to send it to the users.

## 5.9 Generate Teams ( Janith Deshan – 10676638)

USE CASE Title	Generate Teams	
Primary Actor	AI System Administrator (supportive actor)	
Scope	Team R US System	
Level	Summary	
Preconditions	1.Project descriptions and talent pool profiles (including CV's and MBPT results) are available. 2.AI sub system is operational	
Minimal Guarantees	No teams are formed if key data is missing or inconsistent.	
Success Guarantees	Complete justified teams are generated and displays.	
Trigger	AI Sub-System detects that project description and pool member data is ready and initiates team generation automatically.	
Main Success Scenario	Step	Action
	1	AI Sub-System detects complete input data project briefs and user profiles.
	2	AI processes and analyzes project descriptions like skills, duration, etc.
	3	AI analyzes talent profiles skills, experience, MBPT types and CV's.
	4	AI matches users to projects and forms balanced teams.
	5	AI generates justification for each team member's selection.
	6	Teams and justifications are stored and displayed to the Administrator.
	7	Administrator can review and accept or reassign members.
Extensions	Step	Branching Action

	1a	MBPT or CV data missing system flags users as “incomplete” and excludes from generation.
	4a	Skills mismatch between users and project and AI highlights weak matches in justification.
	7a	Admin disagrees with team composition to manually reassignment feature is enabled
Sub-Variations		Branching Action
	1a	<b>MBPT or CV data missing</b>  1: System displays a list of incomplete profiles 2: Admin is given the option to exclude them or wait for data
	4a	<b>Skill mismatch</b>  1: AI flags team as low-compatibility 2: Suggests best-available matches or prompts admin to review manually
	7a	<b>Admin disagrees with generated teams</b>  1: Admin manually removes a member from a team 2: Admin drags and reassigns user to another team 3: System revalidates the team balance and updates justification

## 5.10 Edit Talent Pool (Janith Deshan – 10676638)

USE CASE Title	Edit Talent Pool	
Primary Actor	System Administrator	
Scope	Team R US System	
Level	Primary Task	
Preconditions	1. Administrator is authenticated through secure login. 2. Talent Pool is already created in the system.	
Minimal Guarantees	System remains unchanged if operation is cancelled or fails validation.	
Success Guarantees	Talent pool member information is updated as requested.	
Trigger	Administrator selects “Edit Talent Pool” from the dashboard.	
Main Success Scenario	Step	Action
	1	Administrator logs into the system successfully.
	2	Administrator navigates to Talent Pool management.
	3	System displays a list of all talent pool members.
	4	Administrator selects a member to edit.
	5	Administrator modifies fields such as skills, MBPT result, or removes member.
	6	Teams and justifications are stored and displayed to the Administrator.
	7	System validates new data entries.
	8	System updates the member's profile and confirms success.
Extensions	Step	Branching Action
	4a	Member ID not found while system displays an error
	6a	Invalid format or missing mandatory fields System highlights issues and requests correction.
	7a	Administrator cancels the update → System discards unsaved changes and returns to member list.

Sub-Variations		Branching Action
	4a	<b>Member ID not found</b>  1. System displays “Member not found” and returns to member list.
	6a	<b>Invalid format or missing fields</b>  1: System highlights invalid entries and shows error message. 2: Admin re-enters correct values and proceeds.
	7a	<b>Administrator cancels the update</b>  1: System prompts “Discard unsaved changes?” 2: Admin selects “Yes” → returns to dashboard 3: Admin selects “No” → remains on edit form

## 6. Appendix 2

### 6.1 Complete Application Prototype

#### System Login

**Teams R Us**  
Intelligent Team Formation & Project Management System

[Login](#)
[Admin Dashboard](#)
[Project Management](#)
[Talent Pool](#)
[Member Portal](#)

**Secure Login**

Email Address


Password

Role

[Login to System](#)

Secure authentication with role-based access control

## Administrator Dashboard



### Teams R Us

Intelligent Team Formation & Project Management System

[Login](#)[Admin Dashboard](#)[Project Management](#)[Talent Pool](#)[Member Portal](#)

#### Administrator Dashboard

24  
Active Projects

156  
Talent Pool Members

12  
Teams Formed

89%  
MBPT Completion

##### Quick Actions


Upload New ProjectForm TeamsSend NotificationsGenerate Reports

##### Recent Notifications

**MBPT Test Reminder**  
Sent to 23 members - 2 hours ago

**Team Assignment Complete**  
Project Alpha teams formed - 1 day ago

## Member Portal Dashboard




### Teams R Us

Intelligent Team Formation & Project Management System

[Login](#)[Admin Dashboard](#)[Project Management](#)[Talent Pool](#)[Member Portal](#)

#### Member Portal

##### Upload CV




Upload your CV  
PDF or DOCX format only

Current CV: resume\_2024.pdf

Remove Current CV

##### Personality Test



**Myers-Briggs Personality Test**  
Complete your MBPT to help with team formation

**Status: ENFP - Campaigner**  
Completed on March 15, 2024

Retake Test

##### My Profile

Full Name

Alex Thompson

Email Address

alex.thompson@email.com

Skills

JavaScript, React, Node.js, MongoDB, Project Management, Agile Methodologies

Years of Experience

4

Profile information is managed by administrators. Contact admin for updates.

##### My Notifications

**Team Assignment**  
You've been assigned to Project Alpha team. Check your email for details.  
2 hours ago

**MBPT Reminder**  
Please complete your Myers-Briggs personality test.  
1 day ago

Projects Upload and Management Dashboard

Teams R Us

Intelligent Team Formation & Project Management System

Login

Admin Dashboard

Project Management

Talent Pool

Member Portal

Project Management

Upload Project Description

Drop files here or click to upload

Supports PDF and DOCX formats only

Project Name

Estimated Duration (weeks)

Required Skills

Team Size

Enter project name

12

Java, React, Database Design

3-4 members

Save Project

Cancel

Current Projects

Project Name	Skills Required	Duration	Status	Actions
E-Commerce Platform	React, Node.js, MongoDB	12 weeks	ACTIVE	<div>EditDelete</div>
Mobile App Development	Flutter, Firebase, UI/UX	8 weeks	PLANNING	<div>EditDelete</div>

## Talent Pool Generates From AI

The screenshot displays the 'Teams R Us' Talent Pool Management interface. The header includes the logo and title 'Teams R Us' with the subtitle 'Intelligent Team Formation & Project Management System'. The navigation bar contains links for Login, Admin Dashboard, Project Management, Talent Pool (active), and Member Portal. The main section is titled 'Talent Pool Management' and features an 'Add New Member' form. This form includes input fields for Full Name (John Smith), Email Address (john.smith@example.com), Skills (Java, Python, Project Management), and Years of Experience (5). It also has dropdown menus for MBPT Status (Not Completed) and MBPT Type (if completed) (Select Type). Below the form are 'Add Member' and 'Cancel' buttons. A 'Current Talent Pool' section follows, with 'Send MBPT Notifications' and 'Export List' buttons. It contains a table with two members: Sarah Johnson and Mike Chen. Each member row includes buttons for 'Edit', 'View CV', and 'Remove'.

Name	Email	Skills	Experience	MBPT Status	Actions
Sarah Johnson	sarah.j@email.com	React, Node.js, UI/UX	3 years	ENFP	<a href="#">Edit</a> <a href="#">View CV</a> <a href="#">Remove</a>
Mike Chen	mike.chen@email.com	Python, ML, Data Analysis	5 years	PENDING	<a href="#">Edit</a> <a href="#">View CV</a> <a href="#">Remove</a>

## 6.2 Complete Application Development Plan

### 6.2.1 Project Phases and the Timeline

#### Phase 1 – Software Requirements Specification (SRS) and Proposal (12 weeks)

This phase focuses on fulfillment of the requirements, project proposal development, and system architecture identification for Teams R Us talent pool management and AI team composition system.

#### Phase 2 – Prototype Development (12 months)

This phase consists of the building, testing, and delivery of the prototype as per the SRS requirements. Throughout this phase, Agile development will be utilized for the development of the Teams R Us system. Development phase would be divided into 4 sprints, each lasting for 2 weeks. Each sprint will have multiple milestones aimed at the delivery of incremental features like GUI development, integration of database support, generation of AI teams, file upload features, user authentication, and notification systems.



There will be a review and retrospective meeting at the completion of each sprint to gather feedback and improve development practices.

### 6.2.2 Team Roles and Responsibilities

Team Member	Role	Responsibilities
Janith Deshan	Performance Testing Engineer	Develops and executes performance testing plans like static numerical requirements testing, dynamic numerical requirements testing, capacity and scalability testing, peak workload condition testing, data backup and recovery testing, and error response time measurement as described in Section 3.2
Methuli Jayawickrama	Software Engineer (Frontend /GUI System Analyst)	Develops the Tkinter-based desktop GUI application, develops administrator and member of the talent pool user interface elements, develops form-based uploading of CVs and personality tests in a user-friendly way, offers navigation mechanisms, and ensures WCAG 2.1 accessibility compliance
Oshadhi Rajapaksha	Quality Assurance Engineer/ Testing Specialist	Conducts internal code review activities, performs final user acceptance testing, performs security review and validation, installs automated test frameworks, validates system integration, and enforces all quality requirements of Section 3.7 and 3.8
Dineth Sandeepa	Project Manager/ System Analyst	Oversees project timeline, coordinates team members, imposes completion of milestones, builds system requirements specs, conducts stakeholder requirements analysis, and oversees the overall software development lifecycle from the gathering of requirements to final delivery
Amie Shonnelli	Software Engineer (Backend /AI Developer )	Develops backend database schema using SQLite, implements AI-powered team generation algorithms, designs CV and project description file upload and processing mechanisms, integrates MBTI personality test functionality, and oversees interactions with outside APIs to perform team building logic

## 6.3 Development Schedule

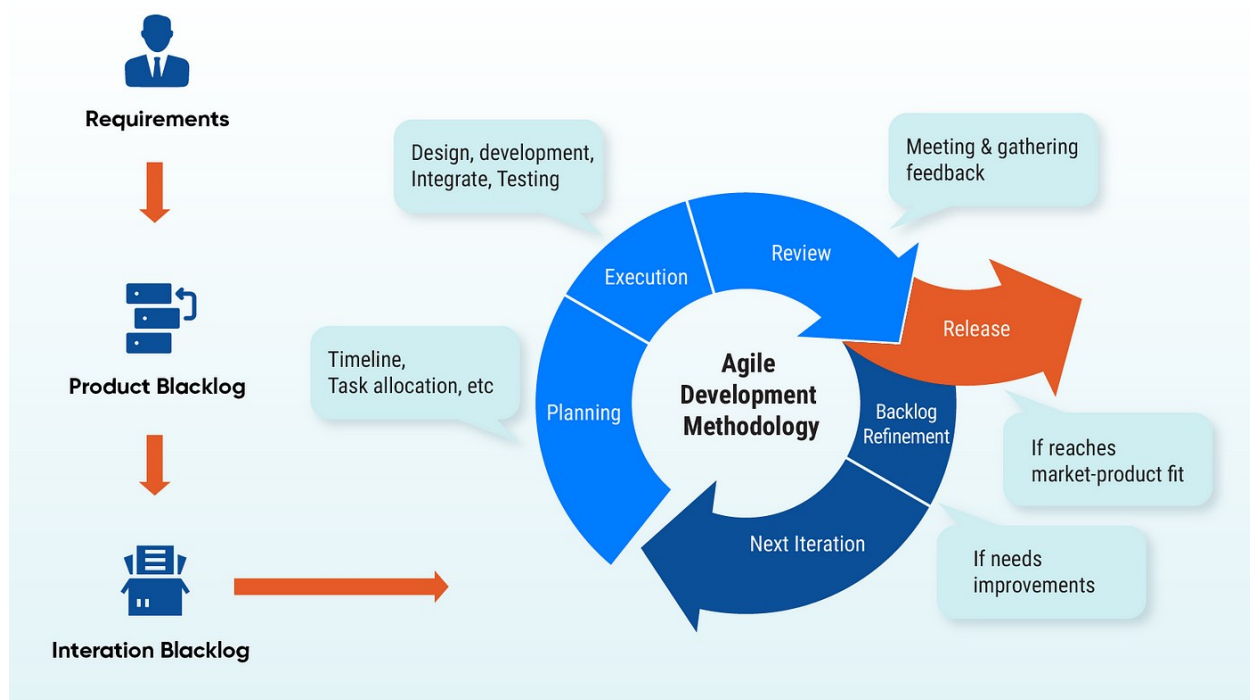
### 6.3.1 Phase 1 – Software Requirements Specification (SRS) – 12 Weeks

Milestone	Description	Deliverable	Timeline	Team Involved
<b>Milestone 1 - Initial Requirements Gathering</b>	Kickoff meeting with unit coordinator and stakeholders to validate the Teams R Us system requirements for AI-enabled team creation and talent pool management	Functional requirements for the administrator and talent pool member interfaces are included in the first edition of the SRS.	Week 1 - 2	Project Manager (Dineth), Software Engineers (Methuli, Amie), Quality Assurance (Oshadhi)
<b>Milestone 2 - High-Level System Design</b>	Design the Teams R Us system architecture comprising Tkinter GUI framework, SQLite database schema, AI team generation algorithms, and file upload mechanisms	System architecture, database ERD, and component diagrams are included in this high-level design document.	Week 3 - 4	Software Engineer (Amie), Project Manager (Dineth)

<b>Milestone 3 - SRS Refinement</b>	Finalize and finalize the SRS incorporating feedback from unit coordinator, such as accurate performance requirements, usability specifications, and interface requirements	All functional and non-functional criteria are included in the final Software Criteria Specification (SRS) document.	Week 5 - 7	Project Manager (Dineth), Software Engineers (Methuli, Amie), Performance Engineer (Janith), Quality Assurance (Oshadhi)
<b>Milestone 4 - Prototype Development Proposal</b>	Develop an in-depth prototype phase plan including sprint planning, technology stack selection, and development process for the 8-weeks implementation period	A comprehensive proposal for the development of a prototype that includes a schedule, resource allocation, and technical specifications	Week 8 - 10	Project Manager (Dineth), Software Engineers (Methuli, Amie)

<b>Milestone 5 - Final Presentation</b>	Present SRS and Prototype Proposal to the unit coordinator for approval, demonstrating in-depth understanding of Teams R Us system requirements and implementation plan	SRS and prototype proposal approved by the client, with official approval to move further to the development phase	Week 11 - 12	Project Manager (Dineth), GUI Developer (Methuli), Quality Assurance (Oshadhi)
-----------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------	--------------	--------------------------------------------------------------------------------

### 6.3.2 Phase 2 – Prototype Development



### 6.3.3 Phase 3 – Prototype Development

Sprint ID	Milestone	Description	Deliverable	Timeline	Team Involved
<b>Sprint 1</b>	Milestone 1 - Sprint Planning	Early work planning on Teams R Us foundation infrastructure for Sprint 1	Sprint backlog for foundation development created	Week 13, Day 1	Project Manager, All Team Members
	Milestone 2 - System Architecture Setup	Set up development environment, version control, and initial structure of Tkinter GUI framework	Development environment configured and basic GUI shell	Week 13, Day 2-3	GUI Developer, Backend Developer
	Milestone 3 - Database	Design and build SQLite database schema for talent	SQLite database	Week 13,	Backend Developer

	Schema Design	pool, projects, teams, and MBTI results	with core tables and relationships	Day 4-5	r, Project Manager
	Milestone 4 - Authentication System	Implement administrator and talent pool member role-based access control and user authentication	Working login system with role separation	Week 14, Day 1-2	Backend Developer, GUI Developer
	Milestone 5 - Basic Navigation Interface	Build main navigation structure and core GUI components for admin and member dashboards	Functional navigation between main system modules	Week 14, Day 3-4	GUI Developer, Quality Assurance
	Milestone 6 - Sprint Review	Review work done in Sprint 1 like authentication, database, and GUI foundation	Feedback gathered and foundation acceptance criteria met	Week 14, Day 5	Project Manager, All Team Members
<b>Sprint 2</b>	Milestone 7 - Sprint Planning	Planning tasks for user admin and file processing functionalities in Sprint 2	Sprint backlog for user management created	Week 15, Day 1	Project Manager, All Team Members
	Milestone 8 - User Profile System	Implement features for creating, editing, and managing talent pool member profiles	Complete user profile management system	Week 15, Day 2-4	Backend Developer, GUI Developer (Methuli)
	Milestone 9 - CV Upload	Implement secure method of uploading files for CVs with PDF and DOCX file support	Working file upload with	Week 15, Day 5 - Week	Backend Developer, Quality

	Implementa tion		validation and storage	16, Day 2	Assuranc e
	Milestone 10 - File Security and Validation	Implement file type validation, size limit, virus scan, and secure storage mechanisms	Comprehen sive file security system operational	Week 16, Day 3- 4	Backend Develope r, Quality Assuranc e
	Milestone 11 - Admin Talent Pool Interface	Implement admin panel for managing talent pools, viewing profiles, and managing members	Admin talent pool manageme nt dashboard	Week 16, Day 4- 5	GUI Develope r, Project Manager
	Milestone 12 - Sprint Review	Review Sprint 2 deliverables like profile system and file upload feature	User manageme nt acceptance criteria validated	Week 16, Day 5	Project Manager, All Team Members
<b>Spri nt 3</b>	Milestone 13 - Sprint Planning	Planning for Sprint 3 work like MBTI integration and AI team generation	Sprint backlog for AI and personality testing created	Week 17, Day 1	Project Manager, All Team Members
	Milestone 14 - MBTI Test Integration	Integrate Myers-Briggs personality test feature with result saving and profile linking	Working personality test system with data persistence	Week 17, Day 2- 4	Backend Develope r, GUI Develope r
	Milestone 15 - Project Description Processing	Develop project description upload and parsing system for requirement and skill extra ction	Project parsing system with requirement extraction	Week 17, Day 5 - Week	Backend Develope r, Performa

				18, Day 1	nce Engineer
	Milestone 16 - AI Team Generation Algorithm	Develop AI-based team formation logic with skill, personality, and project requirement as drivers	Core team generation algorithm with optimization	Week 18, Day 2- 3	Backend Developer, Performance Engineer
	Milestone 17 - Team Assignment Interface	Develop interface for displaying team assignments with explanations and manual adjustment options	Team assignment dashboard with justification display	Week 18, Day 4	GUI Developer, Backend Developer
	Milestone 18 - Algorithm Testing and Optimization	Test AI algorithm performance and fine-tune for accuracy and speed demands	Validated and optimized team generation system	Week 18, Day 5	Performance Engineer, Quality Assurance
	Milestone 19 - Sprint Review	Check Sprint 3 deliverables like MBTI integration and AI team generation feature	AI and personality testing acceptance criteria met	Week 18, Day 5	Project Manager, All Team Members
<b>Sprint 4</b>	Milestone 20 - Sprint Planning	Planning for Sprint 4 activities with focus on end- to-end testing, integration, and final delivery	Sprint backlog for testing and integration created	Week 19, Day 1	Project Manager, All Team Members
	Milestone 21 - System Integration Testing	Conduct rigorous integration testing across all the modules and user flows	Integration testing report with issue	Week 19, Day 2- 3	Quality Assurance, All



			identification		Team Members
	Milestone 22 - Performance Validation	Conduct performance testing to make sure all Section 3.2 requirements like response times and load handling	Performance testing results meeting all specifications	Week 19, Day 3-4	Performance Engineer, Quality Assurance
	Milestone 23 - Internal Code Review	Perform systematic code review processes covering security, quality, and documentation standards	Code review documentation with approval signatures	Week 19, Day 4-5	Quality Assurance, All Team Members
	Milestone 24 - User Acceptance Testing	Conduct user acceptance testing with live users acting as administrators and members of the talent pool	User acceptance testing results with feedback analysis	Week 20, Day 1-2	Quality Assurance, GUI Developer
	Milestone 25 - Bug Fixes and Refinements	Fix issues found, implement bug fixes, and perform final system adjustments	Final system with all critical issues resolved	Week 20, Day 3	All Team Members
	Milestone 26 - Documentation and Demo Preparation	Complete user guides, technical manuals, and 5-minute tutorial video	Complete documentation package and demonstration video	Week 20, Day 4	Project Manager, GUI Developer
	Milestone 27 - Final Review and Delivery	Final system test, stakeholder presentation, and handing over project with peer review	Client-approved Teams R Us prototype with	Week 20, Day 5	Project Manager, All Team Members

			complete deliverables		
--	--	--	-----------------------	--	--

## 6.4 Budget Breakdown

### 6.4.1 Cost estimation of Labour

Sprint	Milestone	Timeline	Team Involved	Hours Estimated	Rate (\$/hr)	Cost (\$)
<b>Sprint 1</b>	Milestone 1 - Sprint Planning	Week 13, Day 1	Project Manager, All Team Members	35	60	2,275
	Milestone 2 - System Architecture Setup	Week 13, Day 2-3	GUI Developer, Backend Developer	65	70	4,550
	Milestone 3 - Database Schema Design	Week 13, Day 4-5	Backend Developer, Project Manager	55	70	3,850
	Milestone 4 - Authentication System	Week 14, Day 1-2	Backend Developer, GUI Developer	80	70	5,600
	Milestone 5 - Basic Navigation Interface	Week 14, Day 3-4	GUI Developer, QA Engineer	50	67	3,350
	Milestone 6 - Sprint Review	Week 14, Day 5	Project Manager, All Team Members	25	65	1,625
	<b><u>Sprint 1 Total 21,250</u></b>					

<b>Sprint 2</b>	Milestone 7 - Sprint Planning	Week 15, Day 1	Project Manager, All Team Members	35	65	2,275
	Milestone 8 - User Profile System	Week 15, Day 2-4	Backend Developer, GUI Developer	90	70	6,300
	Milestone 9 - CV Upload Implementation	Week 15, Day 5 - Week 16, Day 2	Backend Developer, QA Engineer	85	68	5,780
	Milestone 10 - File Security and Validation	Week 16, Day 3-4	Backend Developer, QA Engineer	65	68	4,420
	Milestone 11 - Admin Talent Pool Interface	Week 16, Day 4-5	GUI Developer, Project Manager	55	67	1,625
	Milestone 12 - Sprint Review	Week 16, Day 5	Project Manager, All Team Members	15	40	600
	<b><u>Sprint 2 Total 24,085</u></b>					
<b>Sprint 3</b>	Milestone 13 - Sprint Planning	Week 17, Day 1	Project Manager, All Team Members	35	65	2,275
	Milestone 14 - MBTI Test Integration	Week 17, Day 2-4	Backend Developer, GUI Developer	105	70	7,350
	Milestone 15 - Project Description Processing	Week 17, Day 5 - Week 18, Day 1	Backend Developer, Performance Engineer	75	70	5,250

	Milestone 16 - AI Team Generation Algorithm	Week 18, Day 2-3	Backend Developer, Performance Engineer	120	70	8,400
	Milestone 17 - Team Assignment Interface	Week 18, Day 4	GUI Developer, Backend Developer	55	70	3,850
	Milestone 18 - Algorithm Testing and Optimization	Week 18, Day 5	Performance Engineer, QA Engineer	60	67	4,020
	Milestone 19 - Sprint Review	Week 18, Day 5	Project Manager, All Team Members	25	65	1,625
<b><u>Sprint 3 Total 32,770</u></b>						
<b>Sprint 4</b>	Milestone 20 - Sprint Planning	Week 19, Day 1	Project Manager, All TeamMembers	35	65	2,275
	Milestone 21 - System Integration Testing	Week 19, Day 2-3	QA Engineer, All Team Members	95	67	6,365
	Milestone 22 - Performance Validation	Week 19, Day 3-4	Performance Engineer, QA Engineer	80	67	5,360
	Milestone 23 - Internal Code Review	Week 19, Day 4-5	QA Engineer, All Team Members	70	67	4,690
	Milestone 24 - User Acceptance Testing	Week 20, Day 1-2	QA Engineer, GUI Developer	85	67	5,695
	Milestone 25 - Bug Fixes and Refinements	Week 20, Day 3	All Team Members	65	68	4,420

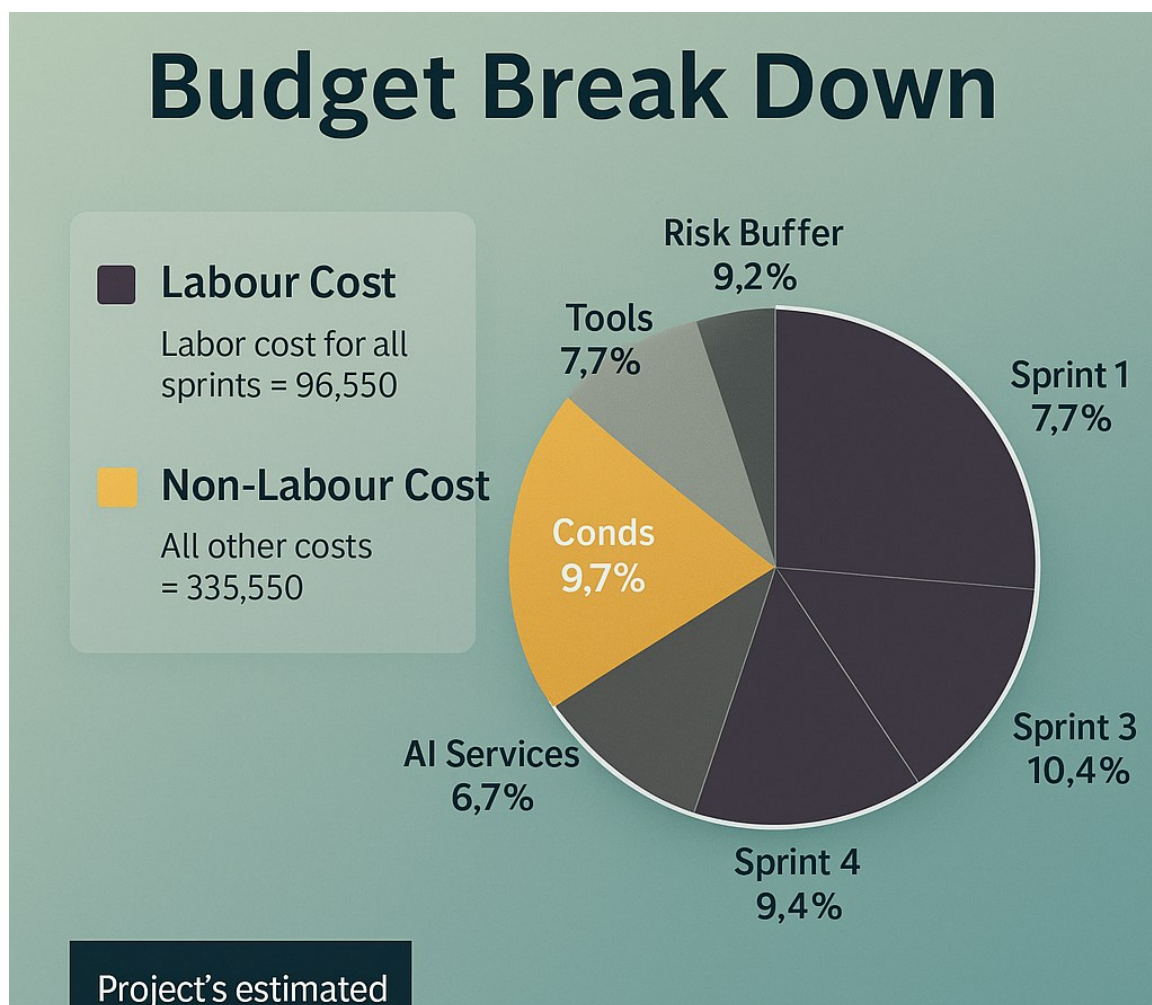
	Milestone 26 - Documentation and Demo Preparation	Week 20, Day 4	Project Manager, GUI Developer	45	66	2,970
	Milestone 27 - Final Review and Delivery	Week 20, Day 5	Project Manager, All Team Members	40	65	2,600
<b><u>Sprint 4 Total 34,375</u></b>						
<b><i>Total Labour Cost for All Sprints: \$112,480</i></b>						

#### 6.4.2 Cost Estimation of Non-Labour Costs

Other Costs	Estimated Cost (\$)
Advanced AI and Machine Learning Platform Services	8,500
Professional MBTI Testing Service Integration and Licensing	6,000
Cloud Infrastructure and File Processing Services (Scalable Storage, Processing)	7,500
Professional Development Tools and IDEs (Enterprise Python, Design Software)	4,500
Hardware and Infrastructure (High-Performance Development Workstations, Testing Servers)	4,000
Hardware and Infrastructure (Development machines, testing equipment)	5,000
Security and Compliance Tools (Enterprise Security Scanning, Vulnerability Assessment)	2,500

Documentation and Presentation Tools (Professional Documentation Suite, Video Production)	2,020
Total Other Costs \$38,020	

#### 6.4.3 Total Project Cost



#### 6.4.4 Total Cost Estimation of the Teams R Us Project

Cost Category	Estimated Cost (\$)
---------------	---------------------

<b>Total Labour Cost (for 4 sprints)</b>	<b>112,480</b>
<b>Advanced AI Services, Enterprise Tools, Professional Infrastructure</b>	<b>38,020</b>
<b>Project Management and Quality Assurance Overhead</b>	<b>12,000</b>
<b>Risk Management and Contingency Buffer</b>	<b>7,500</b>
<b>Total Estimated Project Cost</b>	<b>\$150,000</b>

