**Software Process Selection and Project Plan**

**P05: HR MANAGEMENT SYSTEM**

**<team member names & ids>**

|  |  |
| --- | --- |
| **Student ID** | **Name** |
| **22100289** | **Mohammad Yousuf** |
| **22100260** | **Talha Nasir** |
| **22100230** | **Javeria Tariq** |
| **22100303** | **Ali Adnan Arif** |
| **22100062** | **Aamina Mariam** |

**Table of Contents**

[1.](#_gjdgxs) Introduction 3

[2.](#_30j0zll) Software Process Selection 4

[3.](#_1fob9te) Gantt Chart 5

[4.](#_3znysh7) Risk Management 5

[4.1 Potential Risks and Mitigation Strategies 5](#_2et92p0)

[5.](#_tyjcwt) Development Environment Preparation 6

[6.](#_3dy6vkm) Deployment Platform 6

[7.](#_1t3h5sf) Who Did What? 7

[8.](#_4d34og8) Review checklist 7

# Introduction

The Human resource management system covers many HR aspects from application to management to promotions. The software keeps track of an organization’s employees and provides analytics of their performance using relevant KPIs. The software combines a number of systems and processes to ensure the ease of management in human resources and business processes. The HRMS software helps HR professionals manage the modern workforce.

Our aim is to assist companies in running effectively and efficiently. The system is a suite of software that companies can use to regulate their internal HR functions. Employee data management, recruitment, benefits, training, talent management, employee engagement, and employee attendance include some of the features our software will provide.

# Software Process Selection

**Pros and Cons of waterfall and agile (scrum) processes:**

# The Waterfall Model methodology is followed in a sequential order. The project development team only moves to the next phase of development after the previous step has been completed. The Agile methodology, however, is a practice that implements a continuous iteration of development and testing during the development. The development and testing activities are being done at the same time. The Agile methodology is known for its flexibility whereas Waterfall is a structured software development methodology. Agile performs testing concurrently with software development, whereas in Waterfall methodology testing comes after the “Build” phase. Agile allows changes in project development requirements whereas Waterfall has no scope of changing the requirements once the project development starts.

# Our group will go with the agile model. The waterfall model is not ideal for projects where the requirements are not completely decided at the beginning. That is the case in our project since we will be updating the requirements in the later phases.

**Our Selection:**

# Our project does not have strict regulatory requirements and there is a lot of room to make changes, this pushed us away from the Waterfall model.

**Justification of Selection:**

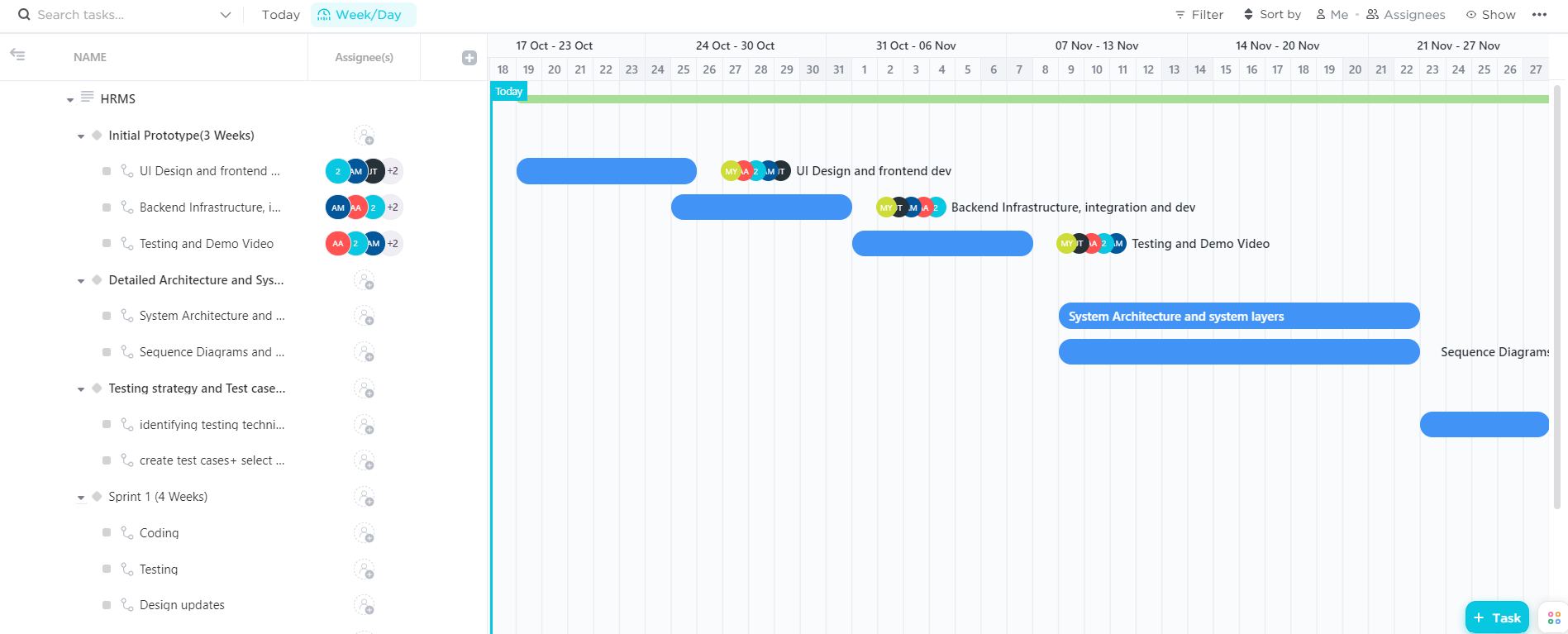
# Since the water model has defined requirements, it leaves less room for creativity which our team would not prefer. Agile however, allows us to adjust requirements and priorities along the way. Our team is trying to implement some innovative features that do not fully exist in any form today so the agile model will be ideal for that.

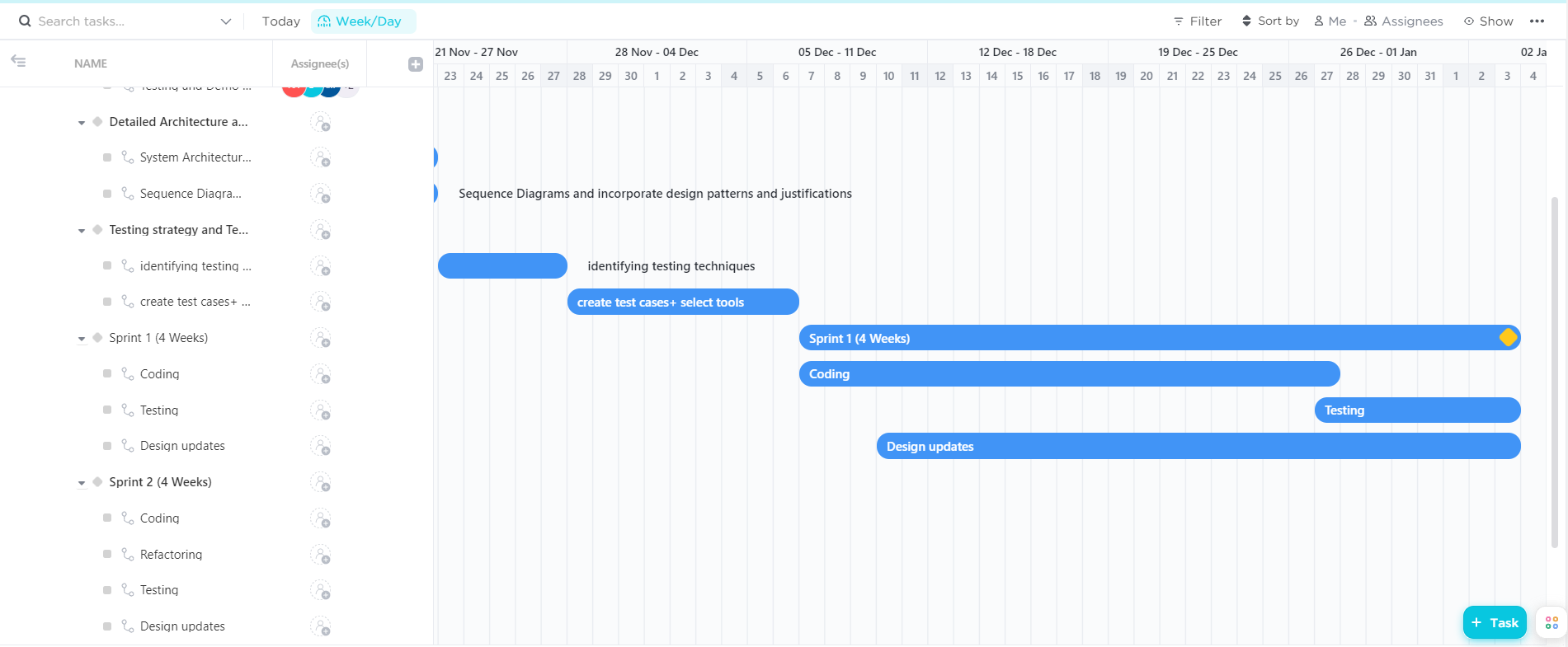
# The major drawback will be the timeline. If the project timeline is fixed and can not be moved, the Waterfall model will offer a more predictable outcome. We however believe that this won’t be that much of an issue due to the great teamwork in our group.

**Project Context Analysis**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1  (Low) | 2 | 3 | 4 | 5  (High) |
| Potential loss due to defect/bugs |  |  | - |  |  |
| Developers’ experience/skills |  |  |  | - |  |
| Rate of requirements change |  |  |  | - |  |
| Team Size (5,10,25,50,100+) | - |  |  |  |  |
| Organization culture (adaptive to change) |  |  | - |  |  |
| Pressure to develop early releases |  |  | - |  |  |
| Business staff’s commitment to work extensively with development team |  |  |  |  | - |
| Developers experience with similar systems |  | - |  |  |  |
| Availability of reusable components |  |  |  | - |  |

# Gantt Chart





# Risk Management

## Potential Risks and Mitigation Strategies

|  |  |  |
| --- | --- | --- |
| **Sr.** | **Risk Description** | **Mitigation Strategy** |
|  | Server failure | There are two ways to fix this issue: to store data in all the accessible databases or distribute it evenly among them |
|  | Broken Authentication | Configure multi-factor authentication whenever possible. The more hoops an attacker has to jump, the harder it is to get into your system. Also paying attention to session management and setting application timeouts correctly. When a user closes a browser, log them out of the system. Any time a user leaves a session while still logged in, the entire system remains vulnerable. |
|  | Sensitive data exposure | Sensitive data exposure can be prevented by   * Encrypting data in transit and at rest. * Using the latest encryption algorithms. * Disabling auto-complete on forms that collect data. * Disabling caching on forms that collect data. |
| 4. | Broken Access Control | It is important to control who has access to which portion of the website. Access control is only effective if it is enforced in trusted server-side code, where the attacker cannot modify the access control authentication.   * Deny access to standard functions. * Use access control lists and role-based authentication mechanisms. * Don't just hide features. |
| 5. | Security Misconfiguration | Security misconfiguration can be mitigated by not allowing default configuration in our website. It can be prevented by   * Disabling administration interfaces * Disabling use of default accounts/passwords. * Configuring the server to prevent unauthorized access, directory listing, etc. * Considering periodically performing scans and audits to help detect future configuration errors or missing fixes. |
| 6. | Cross Site Scripting | In general, effective prevention of cross site scripting vulnerabilities can involve a combination of the following measures:   * **Sanitize User Input**  1. Validate to detect potentially malicious input from users. 2. Encrypt the output to prevent potentially malicious user-supplied data from triggering the autoload and execute behavior by a browser  * **Limit use of user provided data**  1. Only use when it’s necessary 2. Utilize content security policy   By providing additional layers of protection against cross site scripting attacks. |
| 7. | Insufficient logging and monitoring | Depending on the risk of the data being stored or processed by the application:   * Ensure that all server-side login, access control, and access validation errors can be logged with sufficient user context to identify suspicious or malicious accounts and for a sufficient period of time to allow for deferred forensic analysis. * Ensure that high-value transactions have an audit trail with integrity checks to prevent tampering or deletion, such as add-only database tables and the like. * Provide effective monitoring and alerts to detect and address suspicious activity in a timely manner. |
| 8. | Insecure Direct Object References | Perform proper and consistent user authorization and whitelist the choices. More often than not, however, the whole problem can be avoided by storing the data internally. |
| 9. | Unvalidated Redirects and Forwards | This can be prevented by   * Avoiding redirects * Providing a static list of valid locations to redirect to. |
| 10. | Unrestricted File Upload | * Never accept a filename and its extension directly without an allow list filter. * The application should filter and check content for all files uploaded to the server. Files must be carefully analyzed and validated before being made available to other users. When in doubt, the file should be discarded. * It is necessary to have a list of the only allowed extensions on the web application. And the file extension can be selected from the list. |

# Development Environment Preparation

**Tools and Technologies to be used:**

Backend: Django 3.8.2 or 4.0

Frontend: React 17.0

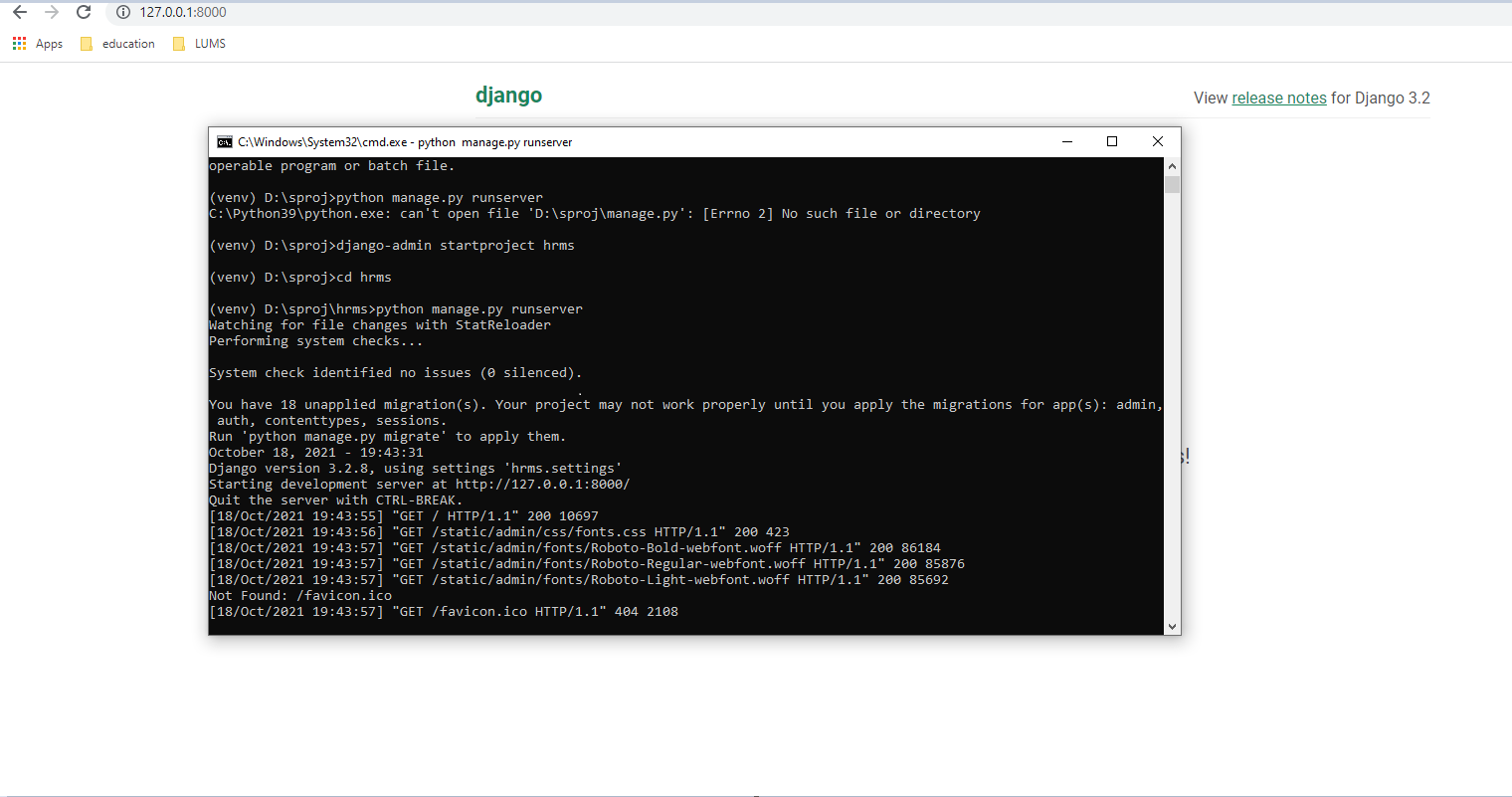
Databases: Amazon DynamoDB or Amazon DocumentDB 3.6

Chatbot Services: AWS Lex V2

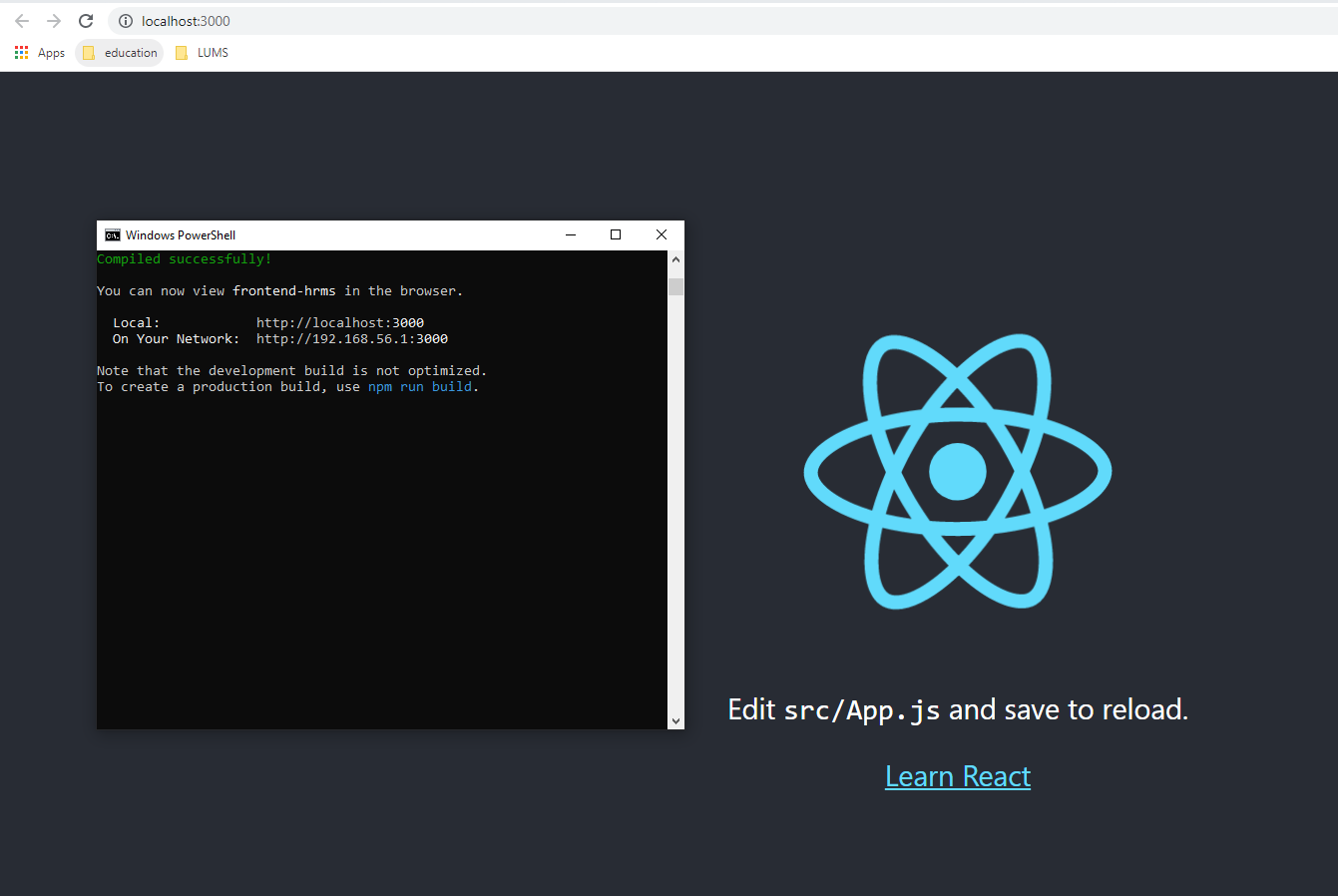
Servers/Serverless: AWS EC2, AWS lambda

We have setup the development environment on our machines and we’ll be using:

**Django for backend**



**React for frontend:**



# Deployment Platform

The deployment platform we decided to use is [Amazon Web Service](https://www.google.com/aclk?sa=L&ai=DChcSEwjlyMz7wczzAhUJgFAGHXcwCAkYABAAGgJkZw&ae=2&sig=AOD64_0x1E9NLYymCORi1X9TXSXVpiLoSQ&q&adurl&ved=2ahUKEwi29cT7wczzAhWi4OAKHYA8CD0Q0Qx6BAgCEAE)s.

This server interface provides access to a wide number of applications and services and is very user-friendly. It also contains a wide variety of tools such as database, software, mobile, analytics, and networking: many of which we plan to use for our project. One major benefit of AWS is also its incredible security and capability to keep your information. Which is one of our major priorities for this project.

# Who Did What?

|  |  |
| --- | --- |
| **Name of the Team Member** | **Tasks done** |
| **Mohammad Yousuf** | Gantt Chart, Development environment prep |
| **Talha Nasir** | Intro, Software Process Selection and Deployment platform |
| **Javeria Tariq** | Risks Management |
| **Ali Adnan Arif** | Risks Management |
| **Aamina Mariam** | Gantt Chart |

# Review checklist

Before submission of this deliverable, the team must perform an internal review. Each team member will review one or more sections of the deliverable.

|  |  |
| --- | --- |
| **Section** **Title** | **Reviewer Name(s)** |
| Section 1&2 | **Ali Adnan Arif, Javeria Tariq** |
| Section 3 | **Talha Nasir** |
| Section 4 | **Aamina Mariam** |
| Section 5&6 | **Mohammad Yousuf** |