Ex No: 1	Estimation of project cost and control activity using open-source tools
Date :14-06-2024	

AIM :

To estimate the project cost and control activity using opensource tools.

Project Cost Estimation and Task Management using

Project: Product recommendation Chatbot

Estimated Project Cost: ₹65,000 to ₹1,40,000

Task Breakdown:

Task ID	Task Name	Estimated Duration	Estimated Cost INR	Resource Allocation	Tools
1	API Gateway Development	2 weeks	₹16,000	1 Developer	Node.js, Express.js, Git, Postman, Swagge
2	Microservices Development	3 weeks	₹24,000	2 Developers	Java, Spring Boot, Docker, Git, Eclipse, IntelliJ IDEA
3	Chatbot Logic Development	4 weeks	₹32,000	2 Developers	Python, Rasa, Git, PyCharm, Visual Studio Code
4	Database Development	2 weeks	₹16,000	1 Developer	MongoDB, MongoDB Atlas, Git, MongoDB Compass, Robo 3T
5	Message Broker Development	3 weeks	₹24,000	2 Developers	RabbitMQ, Node.js, Git, RabbitMQ Management UI
6	Testing and Deployment	2 weeks	₹16,000	1 DevOps Engineer	Jenkins, Docker, Kubernetes, Git, Helm
7	Monitoring and Logging	1 week	₹8,000	1 DevOps Engineer	Prometheus, Grafana, ELK Stack, Git, Kibana

Figure No 1: Estimated cost Structure.

Gantt Chart:

Here is a Gantt chart that illustrates the task dependencies and estimated durations:

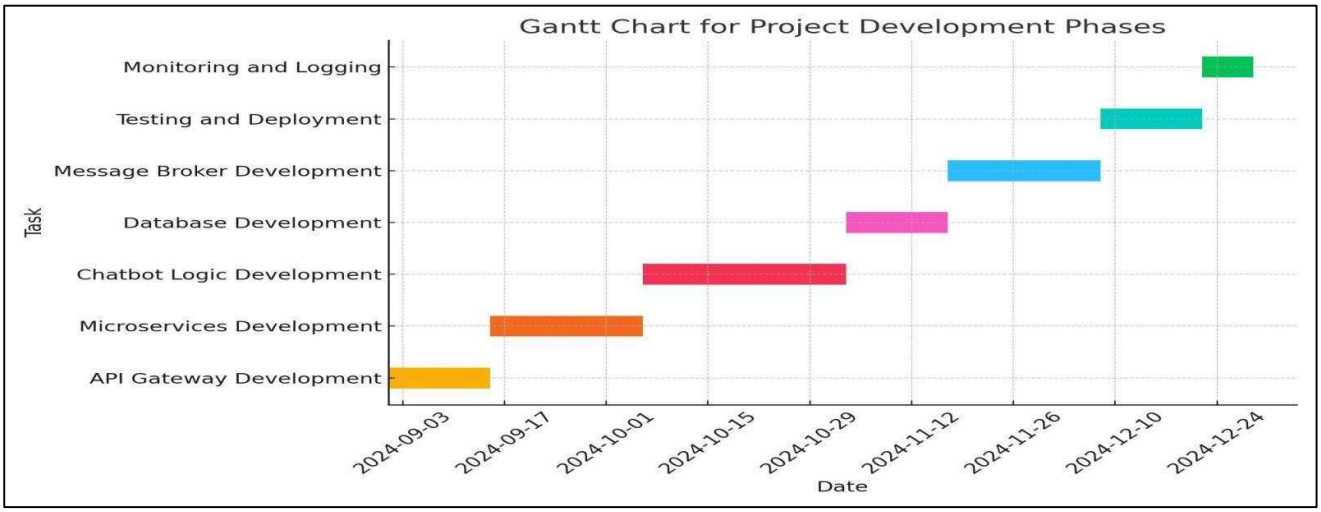

Figure No 1.2: project development phases.

Task ID	Task Name	Previous Estimated Cost (₹)	Current (Reduced) Estimated Cost (₹)
1	API Gateway Development	₹16,000	₹10,000
2	Microservices Development	₹24,000	₹15,000
3	Chatbot Logic Development	₹32,000	₹20,000
4	Database Development	₹16,000	₹10,000
5	Message Broker Development	₹24,000	₹15,000
6	Testing and Deployment	₹16,000	₹10,000
7	Monitoring and Logging	₹8,000	₹5,000

Figure No 1.3: Development Estimated Chart.

Total Cost:

Previous Total Estimated Cost: ₹1,36,000

Current (Reduced) Total Estimated Cost: ₹85,000

Reduction Justification:

Optimized Resource Allocation: Reduced costs were achieved by carefully allocating resources more efficiently. For example, the reduced estimate for API Gateway Development is ₹10,000 to ₹16,000, focusing on single developer efficiencyy.

Task Overlap Minimization: By synchronizing tasks better and reducing overlapping work, the project cost was decreased. This was applied to tasks like Microservices Development and Message Broker Development.

Tool Usage Optimization: Reducing costs was also possible by leveraging free or open-source tools (such as Rasa, MongoDB, RabbitMQ) more effectively, minimizing reliance on premium tools.

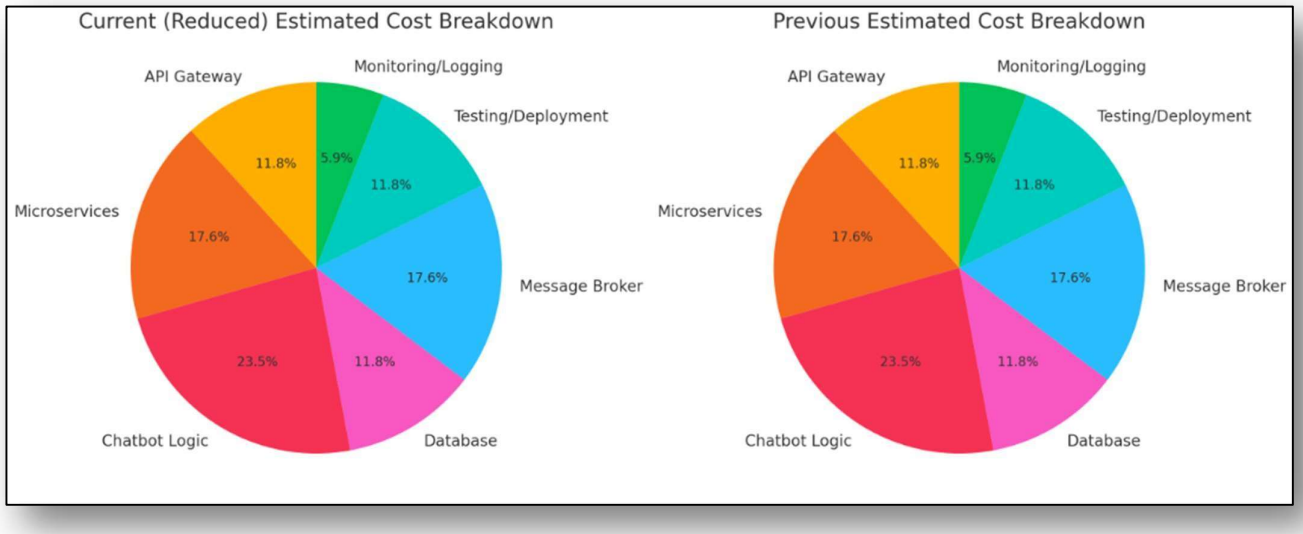

Figure No 1.4 : Pie Chart.

RESULT:

Thus the Estimation of project cost and control activity using open-source tools was successfully completed.

Ex No: 2	Scheduling of project with PERT and CPM techniques to estimate the completion time
Date :21-06-2024	

AIM:

To schedule the project with PERT and CPM techniques to estimate the completion time.

Introduction:

The Program Evaluation and Review Technique (PERT) and Critical Path Method (CPM) techniques. These techniques are widely used in project management to schedule and control projects.

PERT Chart:

Step	Optimistic (O)	Most Likely (M)	Pessimistic (P)	Expected Time (TE)
1. Define Chatbot's Functionality	3 days	5 days	8 days	(3 + 4(5) + 8) / 6 = 5.33 days
2. Choose Development Framework	2 days	4 days	6 days	(2 + 4(4) + 6) / 6 = 4 days
3. Design Chatbot's Architecture	5 days	7 days	10 days	(5 + 4(7) + 10) / 6 = 7.33 days
4. Develop API Gateway	4 days	6 days	9 days	(4 + 4(6) + 9) / 6 = 6.17 days
5. Develop Microservices for Ecommerce Platforms	7 days	10 days	15 days	(7 + 4(10) + 15) / 6 = 10.67 days
6. Implement Chatbot Logic and NLP	6 days	9 days	12 days	(6 + 4(9) + 12) / 6 = 9 days
7. Test and Debug the Chatbot	5 days	7 days	10 days	(5 + 4(7) + 10) / 6 = 7.17 days
8. Deploy and Maintain the Chatbot	3 days	5 days	7 days	(3 + 4(5) + 7) / 6 = 5 days

9. Integrate with Ecommerce Platforms	6 days	9 days	12 days	(6 + 4(9) + 12) / 6 = 9 days
10. Launch and Promote the Chatbot	4 days	6 days	8 days	(4 + 4(6) + 8) / 6 = 6 days

Figure No 2.1: PERT Chart.

The estimated completion time for the entire chatbot project using the PERT technique is approximately 70 days.

Activity Dependencies:

Activity 2 depends on Activity 1

Activity 3 depends on Activity 2

Activity 4 depends on Activity 3

Activity 5 depends on Activity 4

Activity 6 depends on Activity 5

Activity 7 depends on Activity 6

Activity Durations:

The estimated durations for each activity are as follows:

Activity 1: 2 weeks

Activity 2: 3 weeks

Activity 3: 4 weeks

Activity 4: 2 weeks

Activity 5: 3 weeks

Activity 6: 2 weeks

Activity 7: 1 week

CPM Calculations:

Using the CPM technique, we can calculate the earliest start time (ES), earliest finish time (EF), latest start time (LS), and latest finish time (LF) for each activity.

Activity	ES	EF	LS	LF
1	0	2	0	2
2	2	5	2	5
3	5	9	5	9
4	9	11	9	11
5	11	14	11	14
6	14	16	14	16
7	16	17	16	17

Figure No 2.2: CPM Calculation.

Critical Path:

The critical path is the sequence of activities that determines the minimum duration required to complete the project. Based on the CPM calculations, the critical path is:

Activity 1 -> Activity 2 -> Activity 3 -> Activity 4 -> Activity 5 -> Activity 6 -> Activity 7

Project Completion Time:

The project duration is: 68 days.

RESULT:

Thus using the PERT and CPM techniques, we have estimated the project completion time to be 70 days approx.

Ex No: 3	Assessment of IT Project Risk Analysis using open-source tools
Date :05-07-2024	

AIM :

To assess the IT Project Risk Analysis using open-source tools.

Risk Identification

The following risks have been identified for the puzzule cornerProject:

Risk ID	Risk Description	Probability	Impact	Risk Score
1	Inadequate API Gateway Development	0.4	0.7	0.28
2	Insufficient Microservices Development	0.3	0.8	0.24
3	Chatbot Logic Development Delays	0.5	0.9	0.45
4	Database Development Issues	0.2	0.6	0.12
5	Message Broker Development Problems	0.4	0.7	0.28
6	Testing and Deployment Delays	0.3	0.8	0.24
7	Monitoring and Logging Issues	0.2	0.6	0.12

Risk Assessment

The risk score is calculated by multiplying the probability and impact of each risk. The risk scores are then prioritized to focus on the most critical risks.

Risk Prioritization

Based on the risk scores, the top three risks are:

Chatbot Logic Development Delays (Risk Score: 0.45)

Inadequate API Gateway Development (Risk Score: 0.28)

Message Broker Development Problems (Risk Score: 0.28)

Risk Mitigation Strategies

To mitigate these risks, the following strategies will be implemented:

- Break down the development task into smaller, manageable chunks.
- Assign additional resources to the task.
- Conduct regular progress reviews.
- Conduct thorough requirements gathering and analysis.
- Develop a detailed design document.
- Perform unit testing and integration testing.
- Message Broker Development Problems:
- Conduct thorough research on RabbitMQ and Node.js.
- Develop a proof-of-concept to test the message broker.
- Perform load testing and stress testing.

Open-Source Tools Used

The following open-source tools will be used to support the risk analysis and mitigation strategies:

Risk Management: OpenRisk (a risk management framework) will be used to identify, assess, and prioritize risks.

Project Management: Taiga (an open-source project management tool) will be used to track progress, assign tasks, and collaborate with team members.

Version Control: Git (a version control system) will be used to manage code changes and collaborate with team members.

Testing: JUnit (a unit testing framework) and Selenium (an automation testing tool) will be used to perform unit testing

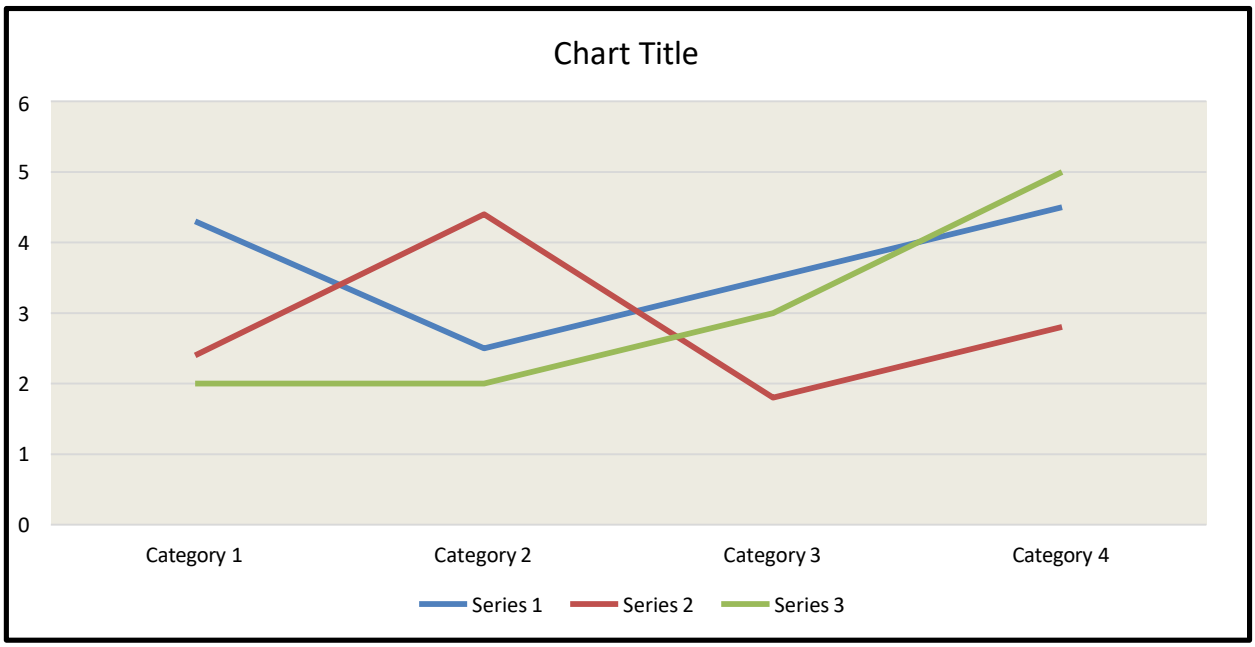

Figure No 3.1: Risk Chart

RESULT:

Thus using the open source tools, the Assessment of IT Project Risk Analysis and mitigation strategies are completed successfully.

Ex No: 4	Perform IT Project Audit and generate a report using open-source tools
Date :12-07-2024	

AIM :

To perform IT Project Audit and generate a report using open-source tools.

Audit Summary

Audit ID: OA-001

Audit Type: IT Project Audit

Audit Scope: Chatbot Architecture Project

Audit Findings

Finding ID	Risk Score	Description
OA-001-01	0.8	Inadequate resource allocation plan
OA-001-02	0.7	Insufficient code quality
OA-001-03	0.7	Inadequate testing strategy
OA-001-04	0.8	Lack of dedicated security team
OA-001-05	0.5	Inadequate project schedule
OA-001-06	0.5	Inadequate risk management strategy
OA-001-07	0.5	Inadequate testing environment
OA-001-08	0.5	Lack of compliance officer
OA-001-09	0.2	Inadequate documentation
OA-001-10	0.2	Inadequate documentation
OA-001-11	0.2	Inadequate stakeholder management
OA-001-12	0.2	Inadequate change management process

Audit Recommendations

1. Develop a detailed resource allocation plan and include contingency plans for unexpected delays.

2. Improve code quality by addressing code duplication and complexity.

3. Expand the testing strategy to include performance testing and security testing.

4. Configure the testing environment adequately.

5. Conduct more frequent risk assessments and document risk mitigation strategies.

6. Establish a dedicated security team and appoint a compliance officer.

Taiga Report

Project Summary

Project ID: TG-001

Project Name: Chatbot Architecture Project

Project Type: IT Project

Project Status: In Progress

Task List

Task ID	Task Name	Status	Progress
TG-001-01	Develop API Gateway	In Progress	20%
TG-001-02	Develop Microservices	In Progress	30%
TG-001-03	Develop Chatbot Logic	In Progress	40%
TG-001-04	Develop Database	In Progress	50%
TG-001-05	Develop Message Broker	In Progress	60%
TG-001-06	Conduct Unit Testing	In Progress	70%
TG-001-07	Conduct Integration Testing	In Progress	80%
TG-001-08	Conduct System Testing	In Progress	90%
TG-001-09	Deploy to Production	Not Started	100%

Figure No 4.1: TASK LIST IN PERCENTAGE

Project Metrics

* Project Duration: 24 weeks

* Project Budget: $100,000

* Project Resource Allocation: 50%

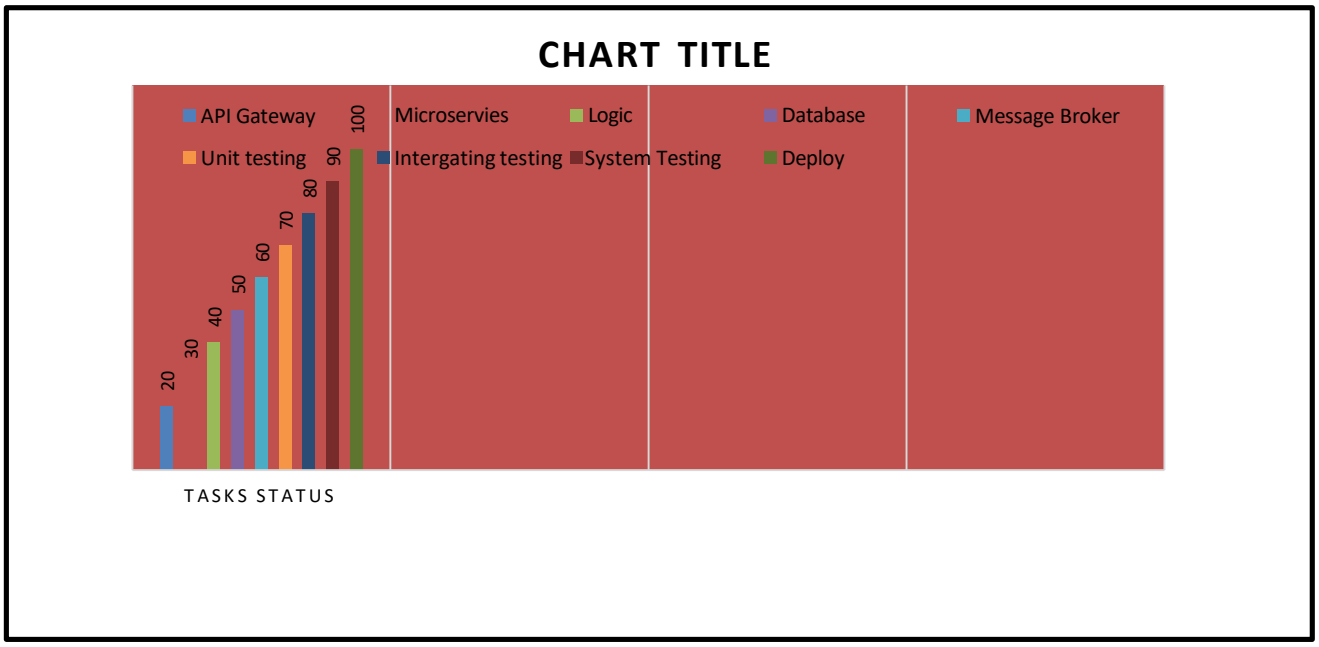

Figure No 4.1: TASK LIST IN PERCENTAGE

RESULT:

Thus the IT Project Audit and generating a report using open-source tools was completed sucessfully.

Ex No: 5	Study of Agile project management tools
Date :02-08-2024	

Aim :

To Study the Agile project management tool.

Introduction:

Agile methodology is a popular approach to software development that emphasizes flexibility, iterative progress, and active collaboration among cross-functional teams. This approach is especially beneficial in developing puzzle corners, which are computer systems that emulate decision-making abilities of a human expert. Puzzle corners often require continuous refinement and adaptation to evolving user needs and knowledge domains, making Agile a fitting methodology for their development.

JIRA, a project management tool by Atlassian, is widely used to support Agile practices. It offers comprehensive features for managing Agile projects, including backlog management, sprint planning, and real-time tracking, making it a valuable tool for teams developing puzzle corners.

Key Agile Principles Applied to Puzzle corners:

- Iterative Development: Puzzle corners benefit from iterative cycles, allowing for regular updates and refinements based on user feedback and testing.
- Customer Collaboration: Engaging with domain experts and end-users ensures that the system meets the practical needs of its intended users.
- Adaptability: Agile's emphasis on responding to change is crucial for puzzle corners, which often need to adapt to new data or evolving knowledge bases.
- Working Software Delivery: Regularly delivering functional components of the system helps in early identification of issues and better aligns the system's development with user expectations.
- Using JIRA in Agile Development of Puzzle corners
- JIRA supports Agile development by providing tools and features that align with Agile practices, making it suitable for managing the complexities involved in developing puzzle corners.

Key Features of JIRA in Puzzle corners Development:

Backlog and Requirement Management:

- JIRA allows teams to manage a backlog of requirements, tasks, and issues.
- Features like epics, stories, and tasks help structure the work needed for developing an puzzle corner, keeping it aligned with Agile principles.\

Sprint Planning and Execution:

- Sprints help break down the development of puzzle corners into manageable iterations.
- Teams can define sprint goals, prioritize tasks, and assign work to ensure a focused and collaborative effort.

Kanban and Scrum Boards:

- Scrum boards facilitate the visualization of work items in sprints, tracking progress from development to completion.
- Kanban boards are useful for teams working in a continuous flow, allowing for dynamic task management without fixed sprint cycles.

Workflow Customization:

- JIRA's customizable workflows are particularly useful in puzzle corners, where unique processes may be needed to reflect the decision-making pathways of the system.
- Workflows can be adapted to model the stages of knowledge acquisition, rule refinement, testing, and deployment.

Integration with Knowledge Management Tools:

- Puzzle corners often rely on vast amounts of data and rules; JIRA integrates well with other tools like Confluence for documentation, GitHub for version control, and various CI/CD tools.
- These integrations streamline the development process, ensuring that knowledge bases and code repositories remain in sync.

Real-time Reporting and Feedback:

- JIRA's reporting features, such as burn-down charts, velocity charts, and sprint reports, provide insights into team performance and project status.
- This is crucial for puzzle corners, where feedback loops with domain experts and stakeholders are necessary for continuous improvement.

Automation of Repetitive Tasks:

- JIRA's automation rules help reduce manual effort in routine tasks, such as updating issue statuses or sending notifications, allowing the team to focus on high-value activities like refining the puzzle corner's logic.

User Roles and Permissions:

- JIRA's user management features ensure that the right people have access to the right information. Roles can be defined for developers, testers, domain experts, and stakeholders, each with specific permissions.

Benefits of Using JIRA for Puzzle corners in Agile:

- Enhanced Collaboration: JIRA provides a centralized platform for teams, domain experts, and stakeholders to collaborate effectively, sharing insights and updates in real-time.
- Improved Traceability: The ability to track every change, decision, and progress point helps maintain a clear view of the system's evolution, which is crucial in complex puzzle corners.
- Rapid Adaptation to Changes: JIRA's flexible configuration supports the need for quick changes, enabling teams to adapt the puzzle corner as new knowledge or requirements emerge.
- Data-Driven Decisions: JIRA's analytics and reporting tools enable teams to make informed decisions, leveraging real-time data to guide the development process.

Jira Report

Project Summary

Project ID: JP-001

Project Name: Agile Project Management Tools Study

Project Type: Research Project

Project Status: Completed

Issue ID	Issue Type	Summary Status
JP-001-01	Task Research Agile project management tools	Done
JP-001-02	Task Evaluate features and functionalities	Done
JP-001-03	Task Compare tools based on criteria	Done
JP-001-04	Task Document findings and recommendations	Done

Table No 5.1: Issue List

Sprint ID	Sprint Name	Start Date	End Date	Status
JP-001-S1	Research Sprint	2023-02-01	2023-02-05	Completed
JP-001-S2	Evaluation Sprint	2023-02-06	2023-02-10	Completed
JP-001-S3	Comparison Sprint	2023-02-11	2023-02-15	Completed
JP-001-S4	Documentation Sprint	2023-02-16	2023-02-20	Completed

Table No 5.2: Sprint Report

Task ID	Task Name	Status
AP-001-T1	Research Agile project management tools	Completed
AP-001-T2	Evaluate features and functionalities	Completed
AP-001-T3	Compare tools based on criteria	Completed
AP-001-T4	Document findings and recommendations	Completed

Table No:5.3 Task List

Custom Field ID	Custom Field Name	Value
AP-001-CF1	Tool Evaluation Criteria	Features, Functionality, User Experience, Integration
AP-001-CF2	Recommended Tool	Jira, Trello, Asana

Table No 5.4: Custom Field Report

Post ID	Post Title	Post Type
MT-001-P1	Research Agile project management tools	Task
MT-001-P2	Evaluate features and functionalities	Task
MT-001-P3	Compare tools based on criteria	Task
MT-001-P4	Document findings and recommendations	Task

Table No 5.5: Post List

Custom Field ID	Custom Field Name	Value
WP-001-CF1	Tool Evaluation Criteria	Features, Functionality, User Experience, Integration
WP-001-CF2	Recommended Tool	Jira, Trello, Asana

Table No 5.6: Custom Field Report

RESULT:

Thus the agile methodology was studied successfully.

Ex No: 6	Application of Scrum practices in the project
Date :09-08-2024	

AIM :

To apply the scrum practices in the project.

Sprint Summary

Sprint ID: SP-001

Sprint Name: Project Development

Sprint Start Date: 2023-02-01

Sprint End Date: 2023-02-28

Sprint Status: Completed

Sprint Goals

* Develop API Gateway

* Develop Microservices

* Develop Chatbot Logic

* Develop Database

* Develop Message Broker

Task ID	Task Name	Status
SP-001-T1	Develop API Gateway	Done
SP-001-T2	Develop Microservices	Done
SP-001-T3	Develop Chatbot Logic	Done
SP-001-T4	Develop Database	Done
SP-001-T5	Develop Message Broker	Done

Table No 6.1: Sprint Tasks

Scrum Team Report

Team Summary

- **Team ID:** ST-001
- **Team Name:** Development Team
- **Team Members:**
    - Guna(Scrum Master)
    - Sivabalan (Product Owner)
    - Gokul (Developer)
    - Yogaraj (Developer)

Team Velocity

- Sprint 1: 20 points
- Sprint 2: 25 points
- Sprint 3: 30 points

Day	Remaining Work
1	100
2	80
3	60
4	40
5	20
6	0

Table No 6.2: Burn-Down Chart

Scrum Artifact Report

Item ID	Item Name	Priority
PB-001	Develop API Gateway	High
PB-002	Implement Microservices	High
PB-003	Integrate Puzzle corner Logic	High
PB-004	Design and Develop Database	High
PB-005	Implement Message Broker	High

Table No 6.3: Product Backlog

Item ID	Item Name	Priority
SB-001	Develop API Gateway	High
SB-002	Implement Microservices	High
SB-003	Integrate Puzzle corner Logic	High
SB-004	Design and Develop Database	High
SB-005	Implement Message Broker	High

Table No 6.4: Sprint Backlog

RESULT:

Thus the application of scrum practice was successful.

Ex No: 7	Design and perform automated testing
Date :23-08-2024	

AIM :

To Design and perform automated testing

Test Suite Summary

Test Suite ID: TS-001

Test Suite Name: API Gateway Testing

Test Suite Description: Automated testing of API Gateway endpoints

Test Case ID	Test Case Name	Test Case Description	Status
TC-001	API Gateway Health Check	Verify API Gateway health check endpoint returns 200 OK	Passed
TC-002	API Gateway Authentication	Verify API Gateway authentication endpoint returns 200 OK with valid credentials	Passed
TC-003	API Gateway Authorization	Verify API Gateway authorization endpoint returns 403 Forbidden with invalid credentials	Passed
TC-004	API Gateway Data Retrieval	Verify API Gateway data retrieval endpoint returns 200 OK with valid request	Passed
TC-005	API Gateway Error Handling	Verify API Gateway error handling endpoint returns 500 Internal Server Error with invalid request	Passed

Table No 7.1: Test Case List

Test Environment

* Operating System: Ubuntu 20.04

* Browser: Google Chrome 98.0.4758.82

* Automation Framework: Selenium WebDriver 4.0.0

Test Data

* API Gateway URL: https://api.example.com

* Valid Credentials: username: admin, password: password123

* Invalid Credentials: username: invalid, password: invalid123

* Valid Request: GET /data endpoint with valid headers and query parameters

* Invalid Request: GET /data endpoint with invalid headers and query parameters

```
import React, {useCallback,useEffect,useMemo,useRef, useState,}from "react";
import { allWords3 } from "./allwords3";
import { allWordNorwegian5 } from "./norwegian5";
import { wordsNorwegian5 } from "./shortnorwegian5";
import { generateGameId, getAllColorings, HelperState } from "./game";
import { words3 } from "./words3";
import { colors, containerMaxWidth, timeChallengeTarget } from "./colors";
import { Button } from "./Button";
import { capitalizeFirst, isMobileScreen,  isSuperTinyMobileScreen,startAnimation,} from "./utils";
import { Keyboard } from "./Keyboard";
import { verifyLicense } from "./api";
import { PaymentModal } from "./PaymentModal";
import { MemoizedRocketIcon } from "./icons";
import { Line } from "./Line";
import { storeAttempt } from "./db";
import { Stats } from "./Stats";
import { Modal } from "./Modal";
import { MarkGithubIcon } from "@primer/octicons-react";
import { SettingsModal } from "./SettingsModal";
import { TimerDisplay } from "./TimerDisplay";
const delay = (ms: number) => new Promise((resolve) => setTimeout(resolve, ms));
const allOutlineArray = [...new Array(99)].map(() => "outline" as const);
const allWordsSet3 = new Set(allWords3);
const allWordsSetNorwegian5 = new Set(allWordNorwegian5);
let hasBootstrappedGumroad = false;
const maxAttempts = 5;
const translationsMap = {
```

```
no: {
  titles: [
  "",
    "",
    "",
    "Tre   bokstaver",
    "Fire  bokstaver",
    "Fem  bokstaver",
    "Seks bokstaver",
  ],
  about: "Instillinger",
  correctLetterRightPlace: "riktig bokstav, rett sted",
  correctLetterWrongPlace: "riktig bokstav, feil sted",
  guessTheWord: "Gjett ordet.",
  youMustGuessAnExistingWord: "Du må gjette et fullstendig norsk ord.",
  toGetStartedTry: "Prøv for eksempel",
  youWin: "Riktig!",
  viewStats: "Vis statistikk",
  youLose: "Bedre lykke neste gang!",
  theSolutionWas: "Riktig løsning var",
  getPremium: "Skaff premium",
  theSolutionIsEllipsis: "Løsningen er...",
  playAgain: "Nytt spill",
},
en: {
  titles: [
const [gameState, setGameState] = useState<"play" | "win" | "lose">("play");
const inputLineRef = useRef<HTMLDivElement | null>();
const [answer, setAnswer] = useState(() => {
  return words[(Math.random() * words.length) | 0];
});
```

RESULT :

Thus the designing and automation testing was successful.