WEEK-01

1(a). Discuss Success and Failure story

Success story of Fampay:

- The app was founded in **2019** by **Ashwin Bhandari** and **Shubham Yadav**. It offers a digital platform that allows teenagers to access financial services, including a prepaid card linked to an app, enabling them to make online and offline purchases while learning financial literacy.
- Fampay's primary target audience is teenagers aged 13-18, a demographic traditionally underserved by financial institutions.
- By catering specifically to this group, Fampay capitalized on a niche market, addressing their unique needs for independence and financial management.
- Fampay offers a prepaid card that can be used for online and offline transactions. This allows teens to experience financial independence while their parents can monitor spending and set limits.
- In addition, the app integrates financial literacy tools, which is essential for educating the younger generation about budgeting, saving, and responsible spending.
- As of 2023, the app had millions of users and had expanded its offerings to include features like bill payments and peer-to-peer transfers.
- Fampay has partnered with banks and financial institutions to offer various services, including digital payments and financial products tailored to teenagers.
- The startup's ability to collaborate with established players in the fintech industry has strengthened its credibility and helped faster growth.
- Fampay invested amount is $**47.70 million dollars.** In 2022, the users in Fampay was around10 million.

Success story on Zoom:

- It was launched on the year **2011**, by the founder named **Eric Yuan**.
- The platform's ability to host video conferences with high-quality audio and video, along with features like screen sharing, breakout rooms, and simple integrations with other platforms, made it a favorite.
- Zoom grew to prominence during the COVID-19 pandemic when video communication became essential for work, school, and social interaction.
- Originally designed as a video conferencing tool for businesses, Zoom became the go-to platform for virtual meetings, webinars, and socializing.
- Zoom also allowed users to join meetings without needing to create an account, which removed barriers to entry.
- In April 2020 alone, the platform saw daily meeting participants jump from 10 million to over 200 million, as millions of people relied on it for everything from business meetings to social gatherings.
- **Simplicity**: Zoom's easy-to-use interface made it accessible to individuals, businesses, and educators.
- **Scalability**: Zoom could handle everything from one-on-one meetings to large conferences, making it suitable for a wide range of users.
- **Reliability**: Zoom's video and audio quality were reliable, which helped build trust in the platform during an era of rapid remote communication.

Failure story on Classmates:

- Classmates.com, a social media platform was founded by **Randy Conrads** and it was launched in **1995**, aimed to connect individuals with old school friends by allowing users to search for and reconnect with classmates from various educational institutions.
- Initially, classmates.com was highly successful, reaching millions of users, failed it had a unique value proposition-reconnecting with long-lost school friends.
- As platforms like Facebook and LinkedIn offered more engaging, interactive experiences, classmates.com became seen as a site primarily for reconnecting with old friends, rather than a dynamic social network.
- People no longer needed a separate platform to reconnect with schoolmates when they could do so through Facebook's much broader network of friends and groups.
- Classmates.com failed to launch an effective mobile version or app at the right time. By the time it entered the mobile app space, social media had already shifted toward more mobile-friendly platforms, with Facebook leading the way. Classmates could not replicate the engaging mobile experiences that platforms like Instagram or Snapchat offered.
- As the digital space grew, so did the number of platforms focused on specific interests or communities, such as LinkedIn for professional networking or Instagram for sharing photos. These platforms began offering more targeted and relevant user experiences, which led to Classmates becoming increasingly marginalized.

Failure story of Yumist:

- It was launched on the year **2014** ,by the founders named **Alok Jain and Abhimanyu Maheshwari**.
- The food delivery market in India was and still is highly competitive.
- The Indian food delivery market was highly competitive from the start, with big players like **Zomato**, **Swiggy**, **Foodpanda**, and others dominating the space.
- These competitors offered a wide range of food choices from numerous restaurants, whereas Yumist only offered a limited selection of home-cooked meals.
- This narrow focus made it hard for the app to stand out. Consumers, especially in urban centers, were more inclined toward variety and quick delivery, which platforms like Swiggy provided.
- It didn't have the financial backing to compete with companies that had more significant funding and larger teams.
- Managing food supply chains, ensuring consistency in meal quality, and maintaining timely deliveries in a country with diverse culinary tastes proved to be a huge operational challenge.
- Yumist faced difficulty with its tech infrastructure and was not able to quickly adapt to the needs of a growing market.
- Yumist struggled with securing additional funding after its initial phase. While the company raised some venture capital to support its growth, it was not enough to survive in such a competitive and capital-intensive industry.
- The startup ultimately ran out of cash, which led to its shutdown in 2016.

1(c). Enact the importance of ethical practices

Ethical Practice Standards

❖ Each standard progresses through four levels of impact:

Foundation Level

❖ At this level you will:
❖ Take responsibility for your actions.
❖ Act consistently with relevant regulations and law.
❖ Handle personal data and information in a professional manner.
❖ Demonstrate honesty in dealings with others.

Associate Level

❖ At this level you will:
❖ Make responsibility choices about your work, apply principles and values.
❖ Consider the purpose and implication of actions, decisions and people practices for all stakeholders.
❖ Provide explanations and reasons for the choices you make and the advice you provide.
❖ Demonstrate professionalism and consistency in what you say and do in order to build trust.

Chartered Member Level

❖ At this level you will:
❖ Make responsible decisions by considering different ethical perspectives, and finding the best possible way forward for all stakeholders.
❖ Coach and Influence managers and leaders to consider the implication of their decisions on stakeholders.
❖ Challenge decisions and actions which are not ethical, explaining the organization risks.
❖ Encourage transparency in decision-making and communication where possible.

Chartered Fellow Level

❖ At this level you will:
❖ Make responsible decisions by balancing different ethical perspectives, and shape how ethics inform wider decision-making and governance.
❖ Coach and influence senior leaders to consider the ethical impact of their decisions in the short and long-term.
❖ Take a visible lead in solving ethical dilemmas, considering how they will play out beyond the organization.
❖ Surface the unsaid in leadership discussions to enable transparency and improved decision making.

Example of Ethics in software engineering:

1. Public
2. Product
3. Judgment
4. Management
5. Client and Employer
6. Profession
7. Colleagues
8. Self

PUBLIC

- Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest. In particular, software engineers shall, as appropriate:
- Accept full responsibility for their own work.
- Moderate the interests of the software engineer, the employer, the client and the users with the public good.
- Be fair and avoid deception in all statements, particularly public ones, concerning software

PRODUCT

- Software engineers shall ensure that their products and related modifications meet the highest professional standards possible. In particular, software engineers shall, as appropriate:
- Strive for high quality, acceptable cost and a reasonable schedule, ensuring significant trade-offs are clear to and accepted by the employer and the client, and are available for consideration by the user and the public.
- Ensure proper and achievable goals and objectives for any project on which they work or propose.
- Ensure that they are qualified for any project on which they work or propose to work by an appropriate combination of education and training, and experience.
- Work to follow professional standards, when available, that are most appropriate for the task at hand, departing from these only when ethically or technically justified.

JUDGMENT

- Software engineers shall maintain integrity and independence in their professional judgment. In particular, software engineers shall, as appropriate:
- Only endorse documents either prepared under their supervision or within their areas of competence and with which they are in agreement.
- Temper all technical judgments by the need to support and maintain human values.
- Maintain professional objectivity with respect to any software or related documents they are asked to evaluate.
- Disclose to all concerned parties those conflicts of interest that cannot reasonably be avoided or escaped.

MANAGEMENT

- Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance. In particular, those managing or leading software engineers shall, as appropriate:
- Ensure good management for any project on which they work, including effective procedures for promotion of quality and reduction of risk.
- Ensure that software engineers are informed of standards before being held to them.
- Ensure that software engineers know the employer's policies and procedures for protecting passwords, files and information that is confidential to the employer or confidential to others

Client and Employer

- Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest.
- In particular, software engineers shall, as appropriate: Provide service in their areas of competence, being honest and forthright about any limitations of their experience and education.
- Not knowingly use software that is obtained or retained either illegally or unethically.
- Use the property of a client or employer only in ways properly authorized, and with the client's or employer's knowledge and consent.
- Ensure that any document upon which they rely has been approved, when required, by someone authorized to approve it.

Profession

- Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
- Particular, software engineers shall, as appropriate:
- Help develop an organizational environment favorable to acting ethically.
- Promote public knowledge of software engineering.
- Extend software engineering knowledge by appropriate participation in professional organizations, meetings and publications.
- Support, as members of a profession, other software engineers striving to follow this Code.

Colleagues

- Software engineers shall be fair to and supportive of their colleagues. In particular, software engineers shall, as appropriate.
- Encourage colleagues to adhere to this Code.
- Assist colleagues in professional development
- Credit fully the work of others and refrain from taking undue credit.
- Review the work of others in an objective, candid, and properly- documented way.
- Give a fair hearing to the opinions, concerns, or complaints of a colleague.
- Assist colleagues in being fully aware of current standard work practices including policies and procedures for protecting passwords, files and other confidential information, and security measures in general.

Self

- Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession. In particular, software engineers shall continually endeavour to.

- Further their knowledge of developments in the analysis, specification, design, development, maintenance and testing of software and related documents, together with the management of the development process.

- Improve their ability to create safe, reliable, and useful quality software at reasonable cost and within a reasonable time.

- Improve their ability to produce accurate, informative, and well-written documentation.

WEEK-02

Overview of Software Development Life Cycle.

The Software Development Life Cycle (SDLC) refers to a methodology with clearly defined processes for creating high-quality software. in detail, the SDLC methodology focuses on the following phases of software development:

- Requirement
- analysis
- Planning
- Software design such as architectural design
- Software development
- Testing
- Deployment

This article will explain how SDLC works, dive deeper in each of the phases, and provide you with examples to get a better understanding of each phase.

What is the software development life cycle?

SDLC or the Software Development Life Cycle is a process that produces software with the highest quality and lowest cost in the shortest time possible. SDLC provides a well-structured flow of phases that help an organization to quickly produce high-quality software which is well-tested and ready for production use.

The SDLC involves six phases as explained in the introduction. Popular SDLC models include the waterfall model, spiral model, and agile model.

So, how does the Software Development Life Cycle work?

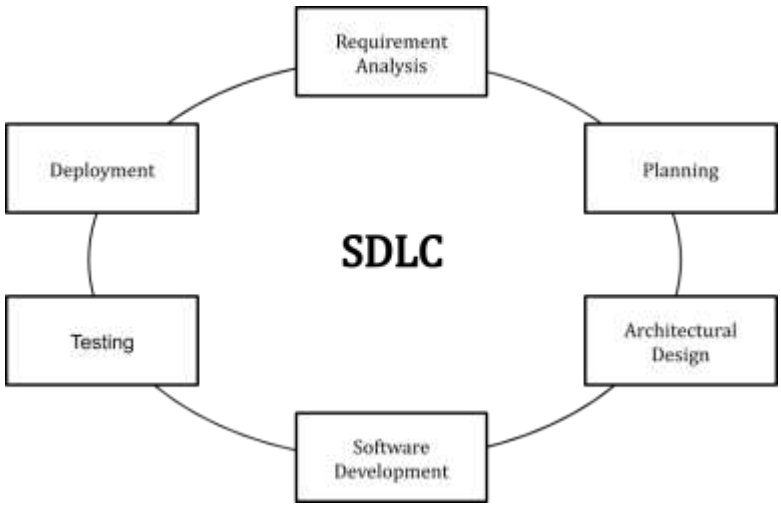
How the SDLC Works

SDLC works by lowering the cost of software development while simultaneously improving quality and shortening production time. SDLC achieves these apparently divergent goals by following a plan that removes the typical pitfalls of software development projects. That plan starts by evaluating existing systems for deficiencies.

Next, it defines the requirements of the new system. It then creates the software through the stages of analysis, planning, design, development, testing, and deployment. By anticipating costly mistakes like failing to ask the end-user or client for feedback, SLDC can eliminate redundant rework and after-the- fact fixes.

It’s also important to know that there is a strong focus on the testing phase. As the SDLC is a repetitive methodology, you have to ensure code quality at every cycle. Many organizations tend to spend few efforts on testing while a stronger focus on testing can save them a lot of rework, time, and money. Be smart and write the right types of tests.

Next, let’s explore the different stages of the Software Development Life Cycle.



Stages and Best Practices

Following the best practices and/or stages of SDLC ensures the process works in a smooth, efficient, and productive way.

1. Identify the Current Problems

"What are the current problems?" This stage of the SDLC means getting input from all stakeholders, including customers, salespeople, industry experts, and programmers. Learn the strengths and weaknesses of the current system with improvement as the goal.

2. Plan

"What do we want?" In this stage of the SDLC, the team determines the cost and resources required for implementing the analyzed requirements. It also details the risks involved and provides sub-plans for softening those risks**.**

In other words, the team should determine the feasibility of the project and how they can implement the project successfully with the lowest risk in mind.

3. Design

"How will we get what we want?" This phase of the SDLC starts by turning the software specifications into a design plan called the Design Specification. All stakeholders then review this plan and offer feedback and suggestions. It's crucial to have a plan for collecting and incorporating stakeholder input into this document. Failure at this stage will almost certainly result in cost overruns at best and the total collapse of the project at worst.

4. Build

"Let's create what we want." At this stage, the actual development starts. It's important that every developer sticks to the agreed blueprint. Also, make sure you have proper guidelines in place about the code style and practices.

For example, define a nomenclature for files or define a variable naming style such as camel Case. This will help your team to produce organized and consistent code that is easier to understand but also to test during the next phase.

5. Code Test

"Did we get what we want?" In this stage, we test for defects and deficiencies. We fix those issues until the product meets the original specifications.

In short, we want to verify if the code meets the defined requirements.

6. Software Deployment

"Let's start using what we got."

At this stage, the goal is to deploy the software to the production environment so users can start using the product. However, many organizations choose to move the product through different deployment environments such as a testing or staging environment

This allows any stakeholders to safely play with the product before releasing it to the market. Besides, this allows any final mistakes to be caught before releasing the product.

Extra: Software Maintenance

"Let's get this closer to what we want." The plan almost never turns out perfect when it meets reality. Further, as conditions in the real world change, we need to update and advance the software to match.

The DevOps movement has changed the SDLC in some ways. Developers are now responsible for more and more steps of the entire development process. We also see the value of shifting left. When development and Ops teams use the same toolset to track performance and pin down defects from inception to the retirement of an application, this provides a common language and faster handoffs between teams.

Application performance monitoring (APM) tools can be used in a development, QA, and production environment. This keeps everyone using the same toolset across the entire development lifecycle.

Examples:
The most common SDLC examples or SDLC models are listed below.

Waterfall Model

This SDLC model is the oldest and most straightforward. With this methodology, we finish one phase and then start the next. Each phase has its own mini-plan and each phase "waterfalls" into the next. The biggest drawback of this model is that small details left incomplete can hold up the entire process.

Agile Model

The Agile SDLC model separates the product into cycles and delivers a working product very quickly. This methodology produces a succession of releases. Testing of each release feeds back info that's incorporated into the next version. According to Robert Half, the drawback of this model is that the heavy emphasis on customer interaction can lead the project in the wrong direction in some cases.

Iterative Model

This SDLC model emphasizes repetition. Developers create a version very quickly and for relatively little cost, then test and improve it through rapid and successive versions. One big disadvantage here is that it can eat up resources fast if left unchecked.

V-Shaped Model

An extension of the waterfall model, this SDLC methodology tests at each stage of development. As with waterfall, this process can run into roadblocks.

Big Bang Model

This high-risk SDLC model throws most of its resources at development and works best for small projects. It lacks the thorough requirements definition stage of the other methods.

Spiral Model

The most flexible of the SDLC models, the spiral model is similar to the iterative model in its emphasis on repetition. The spiral model goes through the planning, design, build and test phases over and over, with gradual improvements at each pass.

1(a). Case Study On SDLC HealthCare Mobile App

❖Overall Abstract of the Healthcare Mobile App

The healthcare mobile app serves as a comprehensive platform that allows patients to track their health, book appointments, connect with doctors via telemedicine, and access medical records. The app also offers reminders for medication, facilitates lab test booking, and provides a secure channel for communication between patients and healthcare providers.

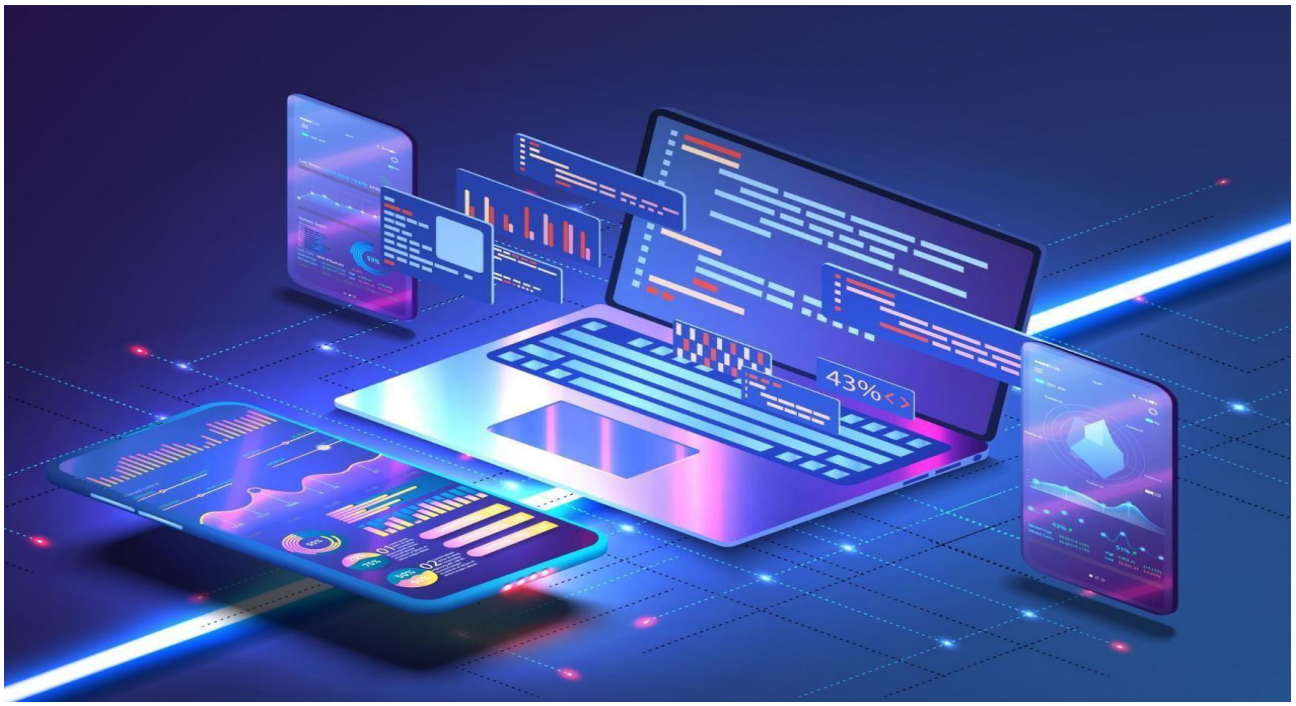
❖Introduction

The healthcare industry is rapidly adopting mobile technology to enhance the delivery of care, improve patient engagement, and increase overall efficiency. This case study explores the Software Development Life Cycle (SDLC) process used to develop a healthcare mobile app designed to improve patient care, manage appointments, monitor health vitals, and offer telemedicine services.

❖SDLC Phases in the Development of the Healthcare MobileApp

Phase 1: Requirement Gathering and Analysis

The first phase involves identifying the needs of the stakeholders (patients, doctors, hospital administrators) and understanding the regulatory compliance, such as HIPAA (Health Insurance Portability and Accountability Act), necessary for data protection in healthcare.



Key Activities:

- Meetings with stakeholders to define the app's goals.
- Research on industry standards and security protocols.
- Gathering functional and non-functional requirements:
    - Patient portal for medical records.
    - Doctor-patient communication (telemedicine).
    - Appointment scheduling.
    - Medication reminders.

Phase 2: Feasibility Study

A feasibility study was conducted to determine the technical, operational, and financial viability of the project. The goal was to identify potential challenges, such as integration with existing healthcare systems (EHR - Electronic Health Records), real-time data processing for vitals monitoring, and ensuring compliance with health regulations.

Key Activities:

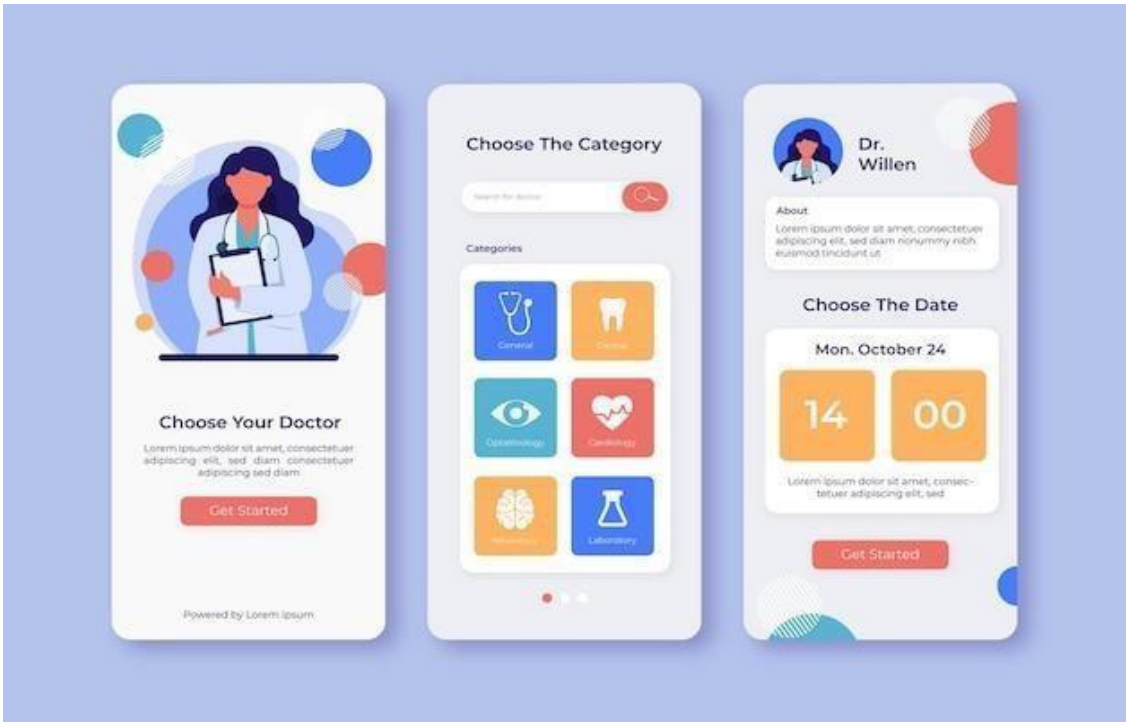
- Analyzing technical feasibility: Evaluating the platforms (Android, iOS) and technologies (React Native, Flutter, Firebase).
- Legal and compliance considerations for data privacy and security.

Phase 3: Design

In this phase, both UI/UX design and system architecture were defined. The app was designed with an easy-to-use interface for patients and healthcare providers, ensuring accessibility for all users.

Key Activities:

- **Database Design**: Database structure was designed to store patient information securely and efficiently. It included personal information, medical history, appointments, prescriptions, etc.
- **API Design**: Developing the architecture to facilitate communication between the app and healthcare databases, including patient records.



Phase 4: Development

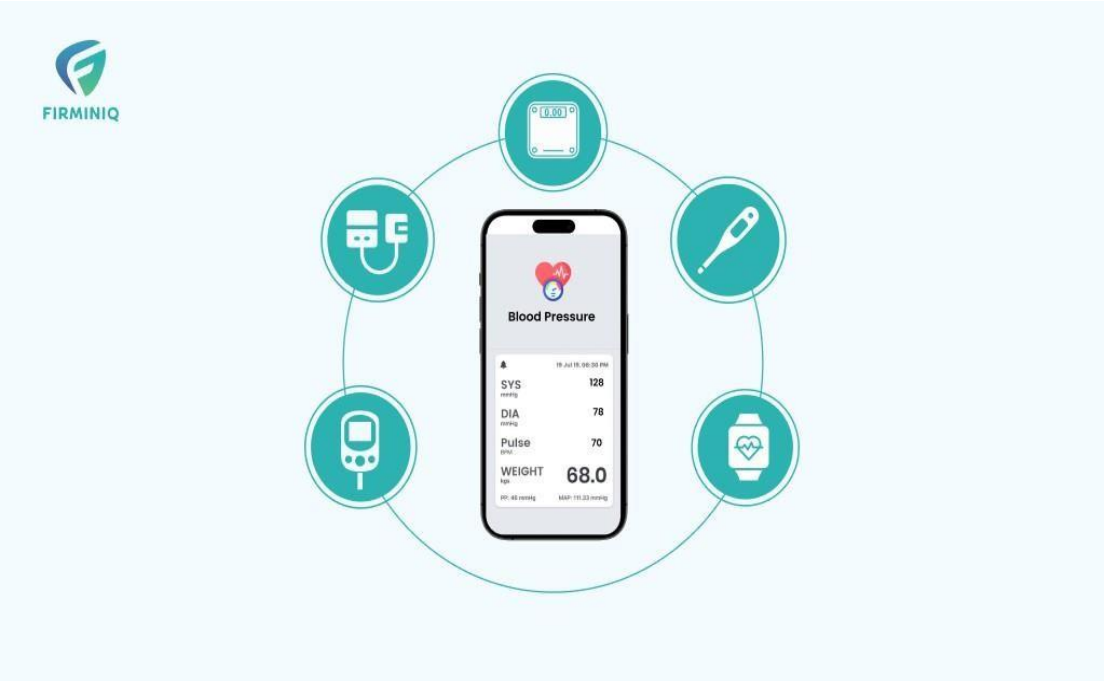
During the development phase, the mobile app was built using agile methodologies, allowing for iterative development and feedback collection. The app was developed for both Android and iOS platforms using React Native.

Key Activities:

- Implementing features like appointment scheduling, and health record management.
- Ensuring secure data transfer and storage using encryption protocols like HTTPS, AES, etc.
- Real-time monitoring and integration with wearable devices for health vitals (e.g., heart rate, blood pressure).

Phase 5: Testing

Testing ensures the app is free of bugs and issues. It includes functional testing, performance testing, security testing, and user acceptance testing (UAT). Special attention was given to data security and ensuring HIPAA compliance.



Key Activities:

- Unit Testing: Ensuring individual functions perform as expected.
- Integration Testing: Verifying the app works seamlessly with healthcare APIs and third- party systems (e.g., EHR systems).
- Security Testing: Evaluating encryption, authentication, and authorization mechanisms.
- Usability Testing: Gathering feedback from a small group of users to assess the app's ease of use and overall experience.

Phase 6: Deployment

Once the testing phase was successfully completed, the app was deployed to the Google Play Store and Apple App Store. Additionally, the app was integrated with healthcare providers' back-end systems and ready for real-time patient engagement.

Key Activities:

- Monitoring app performance post-launch for any bugs or issues.
- Implementing analytics tools to track user behavior and engagement.

Phase 7: Maintenance and Updates

Post-launch, the app went through regular maintenance to address bugs, enhance features, and ensure compliance with the latest healthcare regulations.Regular updates were rolled out to improve the app's functionality and security.

Key Activities:

- Updating the app to comply with new medical regulations and security standards.
- Introducing new features based on user feedback (e.g., adding new appointment types, integrating with more wearable devices).

❖ Technologies Used

- **Programming Languages**: React Native (for cross-platform development), Java, Swift (for native development)
- **Backend Frameworks**: Node.js, Django
- **Database**: Firebase, MongoDB (for scalability and real-time data management)
- **Cloud Services**: AWS, Google Cloud (for cloud storage and computing)
- **APIs**: Integration with EHR systems, wearable devices (e.g., Apple Health, Google Fit)
- **Security**: SSL/TLS encryption, OAuth2.0 for secure authentication

❖ Challenges Encountered

- **Data Integration**: Integrating the mobile app with various existing hospital management systems (EHR, billing systems) posed technical hurdles.
- **User Adoption**: Encouraging patients and healthcare providers to use the mobile app and migrate from traditional methods to digital solutions was a slow process.

❖ Results and Benefits

- **Enhanced Patient Engagement**: The app significantly improved patient engagement by providing easy access to medical records, appointment scheduling, and direct communication with doctors.
- **Cost Reduction**: Streamlined processes such as appointment scheduling, prescription management, and telemedicine consultations reduced operational costs for healthcare providers.
- **Better Health Outcomes**: By tracking health metrics and providing real-time data, the app helped patients better manage chronic conditions like diabetes and hypertension.
- **Increased Efficiency**: The app reduced paperwork and administrative tasks for healthcare providers, allowing more time for patient care.

❖ Models Used:

➢ **Waterfall Model**:

- A traditional approach where development happens in sequential stages, like requirements gathering, design, development, testing, and deployment.

➢ **Agile Model:**

- Highly suited for healthcare apps, as it allows developers to quickly adjust to changing regulations or new healthcare trends.

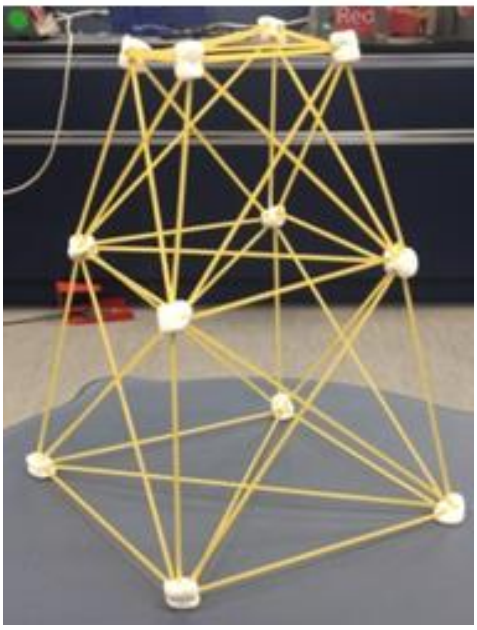
❖ Conclusion:

The SDLC process for the healthcare mobile app was crucial for ensuring its success. Through careful planning, design, and implementation, the app helped bridge the gap between patients and healthcare providers, enabling better patient care and enhanced operational efficiency.

02). Organize and play games to understand the agile process like, Morning wake up game.

(A) The Marshmallow Challenges:

The marshmallow challenge was introduced by Tom Wujec. The Purpose of the challenge is to build the Team Coordination, team Work, Patience, Communication and Time Management.



Objectives:

The goal is to build a freestanding structure using limited materials that can hold a marshmallow without it falling

Requirement Needed for this Challenge Was:

a. 25 sticks of spaghetti
b. 8 Marsh Mallow

- Build the tallest Free-standing Structure in just 20 minutes using no more than 25 sticks of Spaghetti. One yard of String, and 08 Marsh Mallow.
- The structure has to stand firmly on its own, it cannot be propped up, Teams cannot hold on to the Structure when the time runs out. Those touching or supporting the Structure at the end of the exercise will be disqualified.
- The Marshmallow Challenge is a fun and engaging way to build teamwork and problem-solving skills. It's an excellent exercise for corporate teams, students, and anyone looking to boost their collaboration and creativity skills.

(B) White Elephant Sizing:

Overview:

- This game is mainly meant for the analyze of the value from user interface according to the mindset of the team members.
- Group the user stories according to their relative size/efforts taken to build a software.
- Ensure that each teammates gets a chance.
- Learn how user stories are been captured.
- Actively participate in a fun way.
- White Elephant Sizing game mainly is to get the relative size of an agile project and the size of the individual stories before the project starts.
- It mainly gives opportunity to everyone in the team to raise their voice and contribute equally.

Game:

The game is based on food application named Zomato (only an example).

Rules:

If another person in the hall feels wrong can replace the card placed on the board to different column (by giving a solid reason)

Materials:

- Blackboard
- A set of prepared user interface stories.
- Sticky note pads for attaching the related size/value.

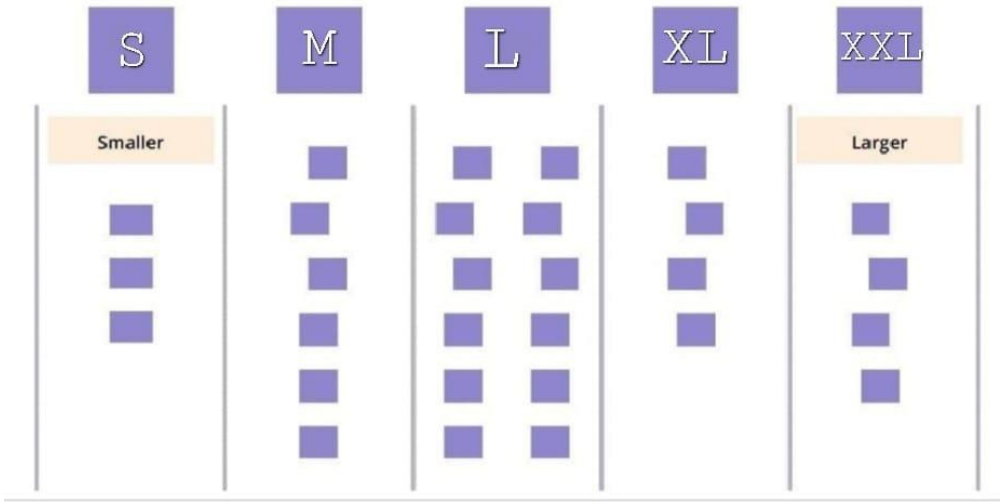


Fig: Elephant sizing

User Interfaces:

- **UI 1:** Logo interface
- **UI 2:** Login /Signup Screen
- **UI 3:** Account details
- **UI 4:** Menu Screen: Displays all food options.
- **UI 5:** Customization Screen
- **UI 6:** Cart Screen
- **UI 7:** Checkout Screen
- **UI 8:** Real-time updates on the order status/Location
- **UI 9:** Settings Screen
- **UI 10:** Offers
- **UI 11:** History

Procedure:

- Create five columns on the board and divide it as S, M, L, XL and XXL.
- The set of prepared user stories on notes. It will be summary with brief description of the different user stories on different notes (tape or sticky notes).
- After the multiple user screens are explained in game.
- After reading the person only has to assign the card to a column. By giving the explanation.
- It will be continued until the user story cards come to an end.

(C) Easter Egg Challenge

A challenge, inspired by this article, that on first sight looks fun and irrelevant, represents in many ways how teams operate, and how that is impacting their productivity. While painting eggs, cutting them out, checking the quality of the delivered product seems extremely easy, the reality shows that even supposedly easy tasks can become difficult and hectic when the participant's mindset is not right.

Requirement needed for this challenge are:

1. A bunch of papers with unpainted easter eggs on them
2. Several boxes of crayons and some scissors

Objectives:

The **challenge** is simple: During the **challenge** the teams will be instructed to deliver the painted **eggs** according to the pre-defined requirements, on time

- Groups of 4 or 5 individuals per team
- Colour and design the 12 unpainted easter eggs.
- After colouring the easter eggs take the scissor and cut in the shape of eggs
- This activity should be finished by 15 minutes
- Then which team has finished the challenge in given time with well-designed eggs will be the winner

03). Create Jira (similar tool) Account and learn Interface

Jira Software:

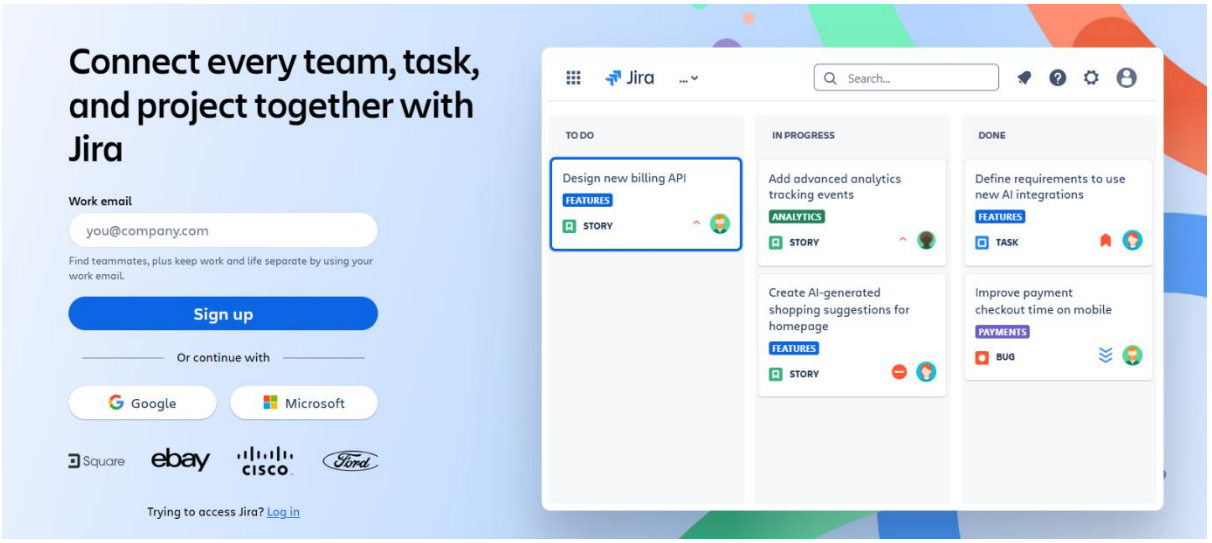
JIRA is a software development tool used for project management and issue tracking. It is a popular tool among software development teams to plan, track, and release software projects. JIRA provides a centralized platform for managing tasks, bugs, and other types of issues and it helps teams to organize and prioritize their work. The tool integrates with other software development tools and has a variety of customizable features and workflows that allow teams to adapt it to their specific needs. Additionally, JIRA also provides various reporting and dashboard features that help teams to stay on top of their work and make Data-driven decisions.

Jira Features:

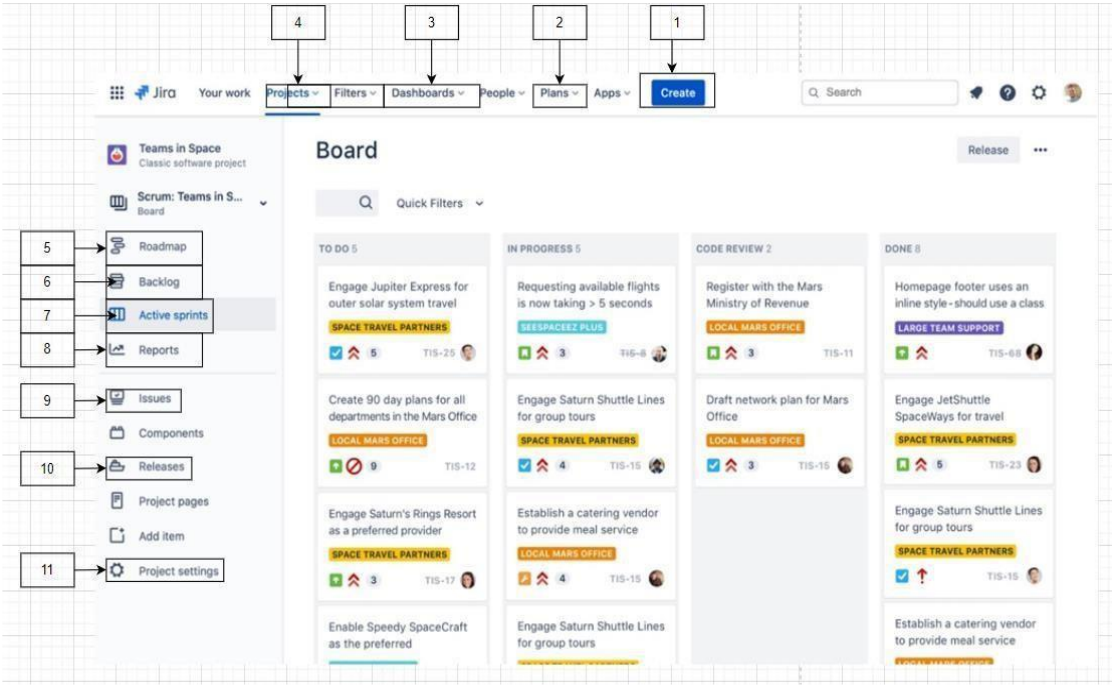
- JQL (Jira Query Language).
- Creates advanced Dashboards.
- Add-ons allows customized features, unique, design.
- Reports.

Creating an Atlassian Account

- Go to the signup page of Atlassian and enter the required user credentials asked on the site and then click on the Sign-Up button.
- To complete the setup and login, click the verification link in the email box.
- Set up your Atlassian account.
- Click on signup after filling details. The account has been created. The user is automatically redirected to the home page.



Interface of Jira Software:



1. Create: You can either create a new epic in agile either use the issues you have created JIRA board

2. Plan: Plan mode display all the user stories create for the project

3. Dashboard: Dashboard is the main display you see when you log in to Jira. You can create multiple dashboards from different projects. OR multiple dashboard one massive overview of all the mark involved with.

4. Project: A project is a simple collection of issues. You would typically you a project to represent the development work for a product, project or services in Jira software,

5. Roadmap: Road map in Jira software or service are team-level road map useful for planning large pieces of work several months in advance at the Epic level within a single project. Dimple planning and dependency management features help your teams visualize and manage work better together.

6. Backlog: The backlog view lists issues that your team plans to work on (and the backlog. Sprint lists) as well as the issues currently your team's board on you use the backlog issue list to plan work in advance so that team members can quickly jump tasks when they're ready.

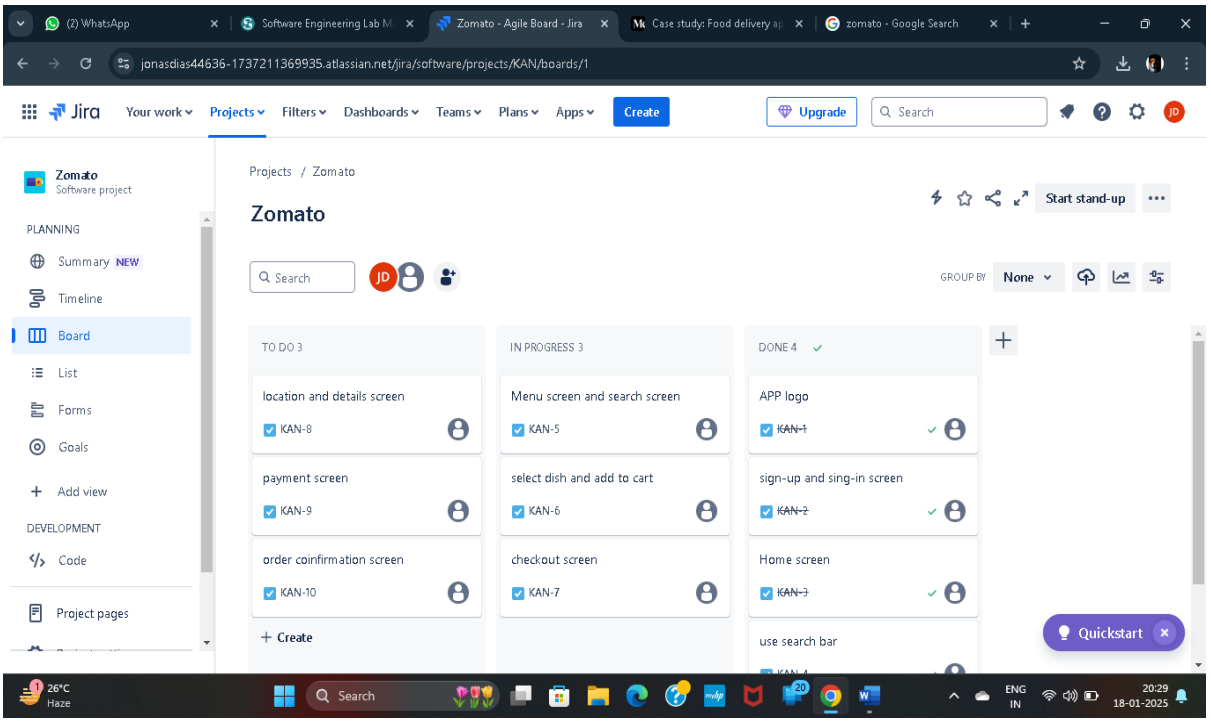
7. Active sprint: The Active sprints of the issues that you're a you're on the most Scrum board displays team is currently working on. You can create and update issues, and drag and drop issues to transition them through a workflow.

8. Reports: Reports in Jira software are offers critical insights for scrum, Kanban, and any agile methodology in between sprint report burn down chart, release burn down, velocity chart. Control charts etc.

9. Issues: A Jira work in issue represents a single piece of a project, Time tracking allows! Teams to record the amount of time they spend working on issues. Teams can create custom workflows to drive the progression of issues.

10. Releases: A scrum or Kanban board. Releases represent paints in time for your project. They can be used to schedule how Features are rolled out to your customers! a way to organize work that has been completed for the project.

11. Project Settings: Once, You have created a project, you can Configure it to suit the needs of your team or to adopt to a new piece of work. Your team Can use Jira projects to coordinate the development product, track a project, manage a help of a desk, and more



WEEK-03

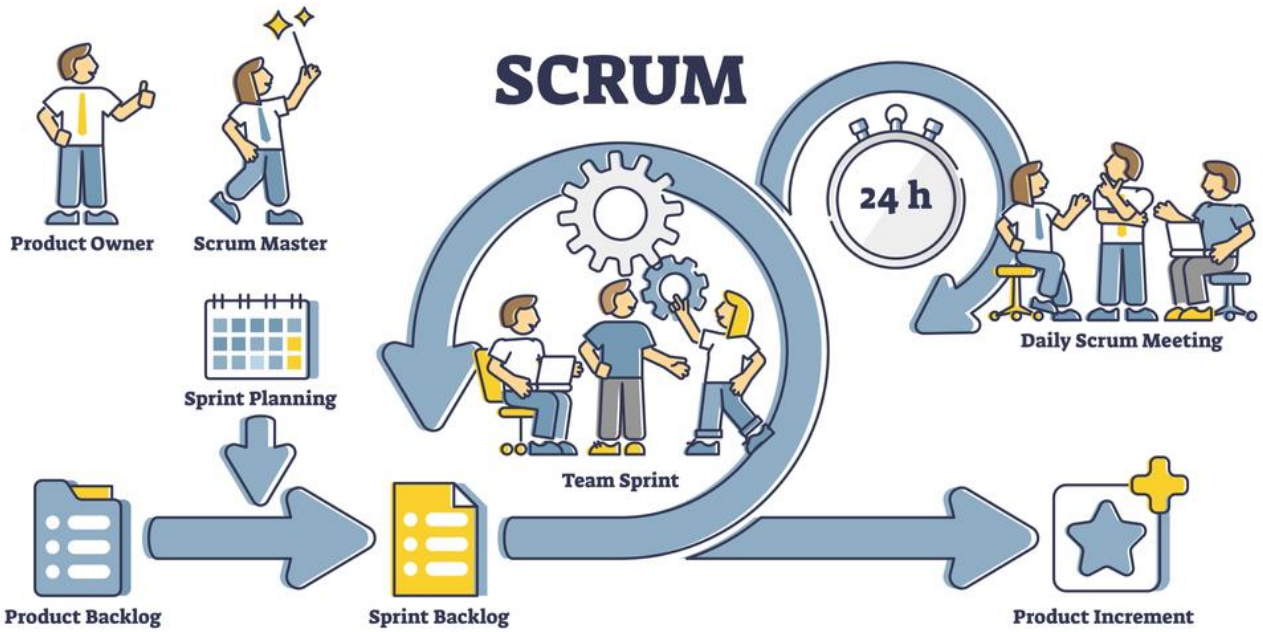
1 a) Play and act agile ceremonies.

Agile Ceremonies On Instagram:

Instagram's development teams often work on features ranging from user interface updates, algorithm changes, content moderation tools, to new functionalities like Stories, Reels, and shopping integrations. Agile ceremonies help ensure that these complex features are developed efficiently, with regular feedback and continuous improvement.

Features of Instagram:

1. Instagram Logo
2. Login Screen
3. Home Screen
4. Story Feature
5. Notification Center
6. Post Interaction
7. Profile Screen
8. Instagram Reels & Shopping Icons
9. Instagram Filters and Effects
10. Color Scheme and Typography
11. Settings and Account Management



➢ Sprint Planning

- Purpose: Define and prioritize the tasks and user stories for the upcoming sprint (usually 1-2 weeks).
- Instagram Context: During Sprint Planning, Instagram's product, engineering, and design teams come together to plan the features or improvements they'll work on in the upcoming sprint. This could include:
  - New filters or effects for Instagram Stories
  - Changes to the Explore page's algorithm
  - Enhancements to user privacy features
- Outcome: A prioritized sprint backlog with clearly defined tasks and a shared understanding of the sprint's goal, ensuring alignment across teams.

➢ Daily Standups (Daily Scrum)

- Purpose: A quick, focused check-in (usually 15 minutes) for each team member to update on progress, roadblocks, and plans.
- Instagram Context: The team gathers daily to discuss:
  - What they worked on yesterday (e.g., finalizing a new feed feature or fixing a bug in the direct messaging system)
  - What they will work on today (e.g., implementing a new photo tagging feature)
- Outcome: Keeps the team aligned, ensures swift identification of problems, and allows for fast resolution of blockers, helping Instagram remain responsive to development needs.

➢ Sprint Review

- Purpose: Review and demonstrate the work completed during the sprint to stakeholders.
- Instagram Context: At the end of each sprint, Instagram's teams demonstrate the features they've built or improved. This could be showcasing a new design for the home feed or a demo of an updated Instagram Stories feature. Stakeholders (such as product managers, marketing, or (leadership) provide feedback based on the demo.
- Outcome: Clear understanding of what's been accomplished, feedback on new features, and alignment on whether the sprint's goals were achieved.

➢ Sprint Retrospective

- Purpose: Reflect on the sprint to identify successes and areas for improvement in processes, tools, or team dynamics.
- Instagram Context: After completing a sprint, the team would gather to discuss:
  - What went well (e.g., faster development cycles, smoother collaboration between engineering and design)
  - What didn't go well (e.g., delays in approval processes, misalignment between design and engineering)
  - What can be improved (e.g., improving communication channels, streamlining testing processes for new features)
- Outcome: Actionable improvements for the next sprint, such as to improve Instagram's development process and team performance.

➢ Backlog Refinement (Grooming)

- Purpose: Regularly review and refine the product backlog to ensure that tasks are ready for the next sprint and are properly prioritized.
- Instagram Context: The product owner, along with key stakeholders, would periodically review and update Instagram's product backlog. This might include:
  - Adding new user stories related to upcoming trends (e.g., new social sharing features or Instagram Shopping updates)
  - Refining user stories based on evolving business goals (e.g., improving Instagram's monetization features or enhancing security)
  - Outcome: A well-prioritized, up-to-date backlog that allows Instagram teams to stay aligned with business goals and quickly respond to user needs and market changes.

Why Agile Ceremonies Matter for Instagram:

- Continuous Improvement: Sprint Retrospectives enable Instagram's teams to reflect and enhance their processes, contributing to faster iteration and delivery of features like Stories, Reels, and Shopping.
- Collaboration: With Agile, cross-functional teams (e.g., product managers, designers, developers) stay aligned through regular ceremonies.
- Faster Delivery: Instagram can quickly deliver small, incremental updates that enhance the user experience. This is particularly important for a platform where trends change rapidly, and users expect constant innovation.

Additional Agile Considerations for Instagram:

- Feature Flags: Instagram could use feature flags in their Agile approach to gradually roll out new features to a subset of users. This allows teams to test new functionalities, like changes to the algorithm or new interactive tools in Stories, without impacting the entire user base.
- Cross-Team Collaboration: Instagram operates with multiple teams (e.g., Stories, Feed, Explore, Ads), so periodic cross-team meetings might be needed to align on shared goals, such as overall platform consistency or cross-functional features (e.g., monetization tools that span across the app).

In conclusion, by adopting Agile ceremonies like Sprint Planning, Daily Standups, Sprint Reviews, Retrospectives, and Backlog Refinement, Instagram ensures that its development process is responsive, efficient, and aligned with its users' needs. These ceremonies provide structure to the fast-paced environment while maintaining flexibility, allowing Instagram to innovate and improve its features continuously.

3(b). Play Different Agile Roles Eg. Product Owner, Business Analyst

The two key pillars for a successful agile project are the Product Owner (PO) and the Business Analyst.

PRODUCT OWNER:

A product owner is a role on a Scrum team that is responsible for the project's outcome.
The product owner seeks to maximize a product's value by managing and making product best and effective.

ROLES AND RESPONSIBILITY OF A PRODUCT OWNER:

1. **Defining the vision:** The agile product owner is the person on the product development team, using their high-level perspective to define goals and create a vision for development projects.
2. **Managing the product backlog:** One of the most important responsibilities for a scrum product owner is managing the product backlog.
3. **Prioritizing needs:** Another key role of the product owner is to prioritize needs. In other words, they must prioritize the triangle of scope, budget, and time, weighing priorities according to the needs and objectives of stakeholders.
4. **Overseeing development stages:** With the vision, strategy, and product priorities set, the product owner should spend a significant amount of time overseeing the actual development of the product. They are a key player throughout each event, including planning, refinement, review, and sprint.

SOFT SKILLS NEEDED TO A PRODUCT OWNER:

- Communication
- Technical Skills.
- Decision-Making.
- Project Management Skill.
- Collaboration

BUSINESS ANALYST:

Agile business analyst is about increasing the delivery of business value to the stakeholders of the project or product being developed.

ROLE OF A BUSINESS ANALYST IN AGILE:

- Maintain the focus on business value
- Identify missing requirements
- Coach the Product Owner (on work)
- Coach the development team (on business domain)
- Help define acceptance criteria before work starts
- Solving Business Problems
- Looking for Savings and Efficiencies
- Focusing on Business Development
- Performance Analysis
- Competitor Analysis

SOFT SKILLS OF A BUSINESS ANALYST ARE:

Analytical thinking and Problem Solving-solving, creative thinking, systems thinking, learning and decision making.

1. **Behavioural Characteristics** -ethics, personal organisation and trustworthiness.
2. **Business Knowledge**-business principles & practice, industry knowledge, organisation knowledge and solution knowledge.
3. **Communication Skills**-oral communication, written communication and teaching.
4. **Interaction Skills-facilitation** & negotiation, leadership & influencing and teamwork.
5. **Software Applications**-general purpose applications and specialised application.

WEEK-04

1 a) Case Study To Understand The Importance Of Risk Management And Mitigation Of Risk On Bigbasket

Bigbasket, India's largest online grocery retailer, has revolutionized the way people shop for daily essentials. However, like any e-commerce business, Bigbasket faces various risks, including operational, financial, technological, and cyber risks. Effective risk management has played a crucial role in ensuring the company's sustainability and growth.

❖ Cybersecurity and Data Privacy Risk

➢ **Risk:**

- The breach included sensitive customer information such as names, email addresses, phone numbers, and order history. This exposed BigBasket customers to the risk of identity theft, phishing attacks, and financial fraud.
- In 2020, BigBasket and other e-commerce platforms in India were under scrutiny due to India's data privacy regulations, which were evolving.

➢ **Risk Mitigation:**

- BigBasket enhanced encryption methods for both stored and transmitted data to ensure that sensitive customer information, such as payment details and personal data, was protected from unauthorized access.
- Continuous monitoring of systems and logs to detect, respond to, and mitigate any suspicious activities more efficiently.
- BigBasket focused on ensuring that any third-party vendors or partners followed robust security and data protection protocols.

❖ Financial Risk

➢ **Risk:**

- One of the notable risks came in **2019**, when BigBasket was focusing on expansion, which resulted in increasing operational costs and heavy investment in technology, supply chain, and logistics.
- In 2019, BigBasket was dealing with these risks in the backdrop of intense competition from other online grocery delivery services like Grofers, Amazon, and traditional retail chains venturing into the online space.

➢ **Risk Mitigation :**

- In **2020**, the company raised additional funding from **Tata Group**, which also helped mitigate financial risk by providing a more stable financial backing.

4(b). How to use tool to manage and mitigate risks [e.g.: logic gate, board]

Logic gate:

It is a software where you keep all the information is in one system (logic gate).it includes policies, goals & priorities.

Or

Logic gate's controls management solutions gives your risk and control owners a break from the disorganized mess of spreadsheet email and documents.

Working:

1. It will work from single central hub.
2. It will according to the industry standard control frameworks (ISCF).

For example:

1. Management wants task 'E" to be done on priority but the employee gets the email after days about priority.

Advantages:

By using logic gates everyone gets information at real time.

1. Emails:
   Logic gates can be used for emails as well.

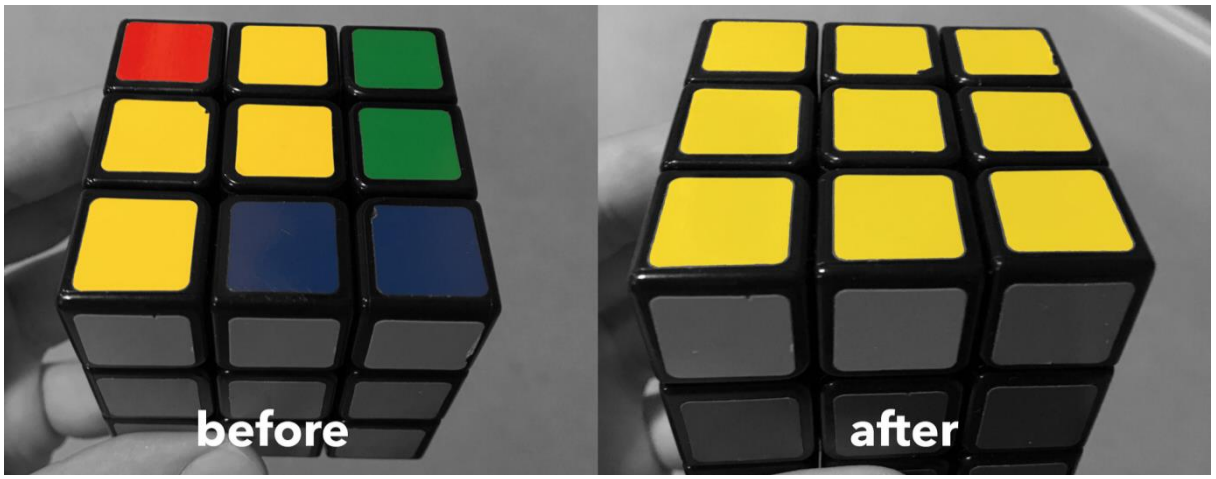
2. If we find any risk then we can plan &everyone can work on real time. It's a rapid session where team will monitor, control assessments, gather evidence and much more, right in the platform. They will continuously collaborate and share information such as tracking, finding or triggering automatic emails when an automatic assessment is kicked off.

Week-05

1). Conduct warmup activities to Ignite Design Thinking.

(a) Arranging Rubik's cube:

Applying **Design Thinking** to solving the Rubik's Cube involves focusing on the user's needs and experience throughout the process. Here's how **Design Thinking** can be applied in **simple steps** to the Rubik's Cube:



❖ Empathize: Understand the User's Experience

- Identify the frustration beginners face with the complexity of solving the cube and their need for clear, simple instructions.

❖ Define: Identify the Core Problem

- The Rubik's Cube is often too challenging for beginners due to complex algorithms and confusion over where to start.

❖ Ideate: Brainstorm Solutions

- Break down the solution into smaller, easier steps, offering visual cues and fewer technical terms to make the process more accessible.

❖ Prototype: Create a Simplified Guide

- Develop a clear, simplified step-by-step guide, such as solving one layer at a time (e.g., Layer-by-Layer method) and using visual aids for each step.

❖ Test: Validate with Users

- Have beginners try the new guide and gather feedback on clarity and ease of understanding.

❖ Refine: Improve Based on Feedback

- Simplify any confusing steps or add clearer instructions based on user input.

❖ Iterate: Make Adjustments

- Update the guide or app interface to reflect the most effective and intuitive way for users to progress through the steps.

❖ Design for Accessibility

- Create a design that caters to different learning styles (visual, kinesthetic, etc.) and provides easy-to-follow steps for various skill levels.

❖ Introduce Gamification

- Add interactive elements like a progress tracker or rewards to motivate users to keep improving.

❖ Launch and Continue to Improve

- Release the guide or app, gather more user feedback, and keep updating it based on users' needs and challenges.

2). Organize and conduct design thinking exercises and games.

(a) Cup Flopping with a Ball:

Here's a simplified Design Thinking game for Cup Flopping with a Ball:

Step 1: Empathize

Understand the challenge: Flip 5 cups using a ball.

Step 2: Define

Define the problem: "How might we flip 5 cups using a ball in the most efficient way possible?"

Step 3: Ideate

Generate ideas:
1. Flick and Flip: Flick the ball to hit the first cup.
2. Roll and Knock: Roll the ball to knock over the cups.
3. Bounce and Flip: Bounce the ball to flip the cups.

Step 4: Prototype

Test your ideas:
1. Start with 2 cups and test your flicking/rolling/bouncing technique.
2. Gradually add more cups, refining your technique.

Step 5: Test

Evaluate your prototype:
1. Measure the success rate of flipping all 5 cups.
2. Refine your technique based on the results.

Step 6: Iterate

Refine and repeat:
1. Adjust your flicking/rolling/bouncing technique.
2. Test again, aiming to improve your success rate.

WEEK-06

1. Organize role play for requirement activities.

Prime video requirements:

1. User Registration/Login
2. Plan selection and Activation
3. Payment processing
4. Content Browsing
5. Search content
6. Streaming content
7. Log out

Role play:

1. **Bhuvana** – A movie buff, always looking for the best content.
2. **Dhanush** – A casual viewer, unsure if Prime Video is worth it.

Dhanush: Hey Bhuvana, I've been thinking about getting Prime Video, but I'm not sure if it's worth it. Do you use it?

Bhuvana: Oh, absolutely! It's one of my go-to streaming platforms. What kind of stuff do you like to watch?

Dhanush: Mostly action movies, some good TV shows, and maybe a few originals. Does Prime have anything great?

Bhuvana: Oh, for sure! If you like action, they have the "Reacher" series—it's packed with suspense and fights. And if you're into spy thrillers, "Tom Clancy's Jack Ryan" is amazing!

Dhanush: Hmm, I've heard of Jack Ryan. What about movies?

Bhuvana: Tons! They have a mix of old classics and new blockbusters. And their original movies are top-notch "Air," about Nike signing Michael Jordan, was really good. Plus, they recently added some Marvel and DC films.

Dhanush: Okay, that sounds cool. But what about shows? Netflix kind of dominates there.

Bhuvana: True, but Prime is catching up! Have you heard of "The Boys"**?** It's a dark, crazy take on superheroes—definitely not your typical Marvel-style stuff. And if you like fantasy, "The Lord of the Rings: The Rings of Power" is a must-watch.

Dhanush: That does sound tempting… but is it worth the price?

Bhuvana: Well, it's part of Amazon Prime, so you also get free shipping, music, and books. Honestly, if you shop on Amazon, it's a no-brainer.

Dhanush: You make a good argument! Alright, I might give it a shot.

Bhuvana: Nice! Oh, and don't forget—some movies are available for rent, but they have a massive free library, so you'll never run out of stuff to watch.

Dhanush: Alright, alright, you've convinced me. Let's go binge-watch something!

Bhuvana: Grinning That's the spirit!

2.Identify a problem and prepare requirement documents or Epics and user stories.

Problem Statement:

Users face difficulty in discovering relevant content on Prime Video due to poor search results, ineffective recommendations, and lack of personalized content discovery options.

Objectives:

Improve the search functionality and recommendation system to enhance content discovery and user satisfaction on Prime Video.

Epic: Prime Video

Description: Prime Video is a streaming service that offers a vast library of movies, TV shows, and original content to entertain and engage its users.

User Story 1: Search and Discover Content

As a user, I want to search for specific movies and TV shows, and discover new content based on my interests.

Tasks:

1. Design and implement the search bar.
2. Integrate with Amazon's content database.
3. Test search and discovery functionality.

User Story 2: Play and Control Video

As a user, I want to play movies and TV shows, and control the video playback experience.

Tasks:

1. Design and implement video controls.
2. Implement video settings functionality.
3. Test video playback and control functionality.

User Story 3: Manage Watchlist and Recommendations

As a user, I want to manage my watchlist and receive personalized recommendations based on my viewing history and preferences.

Tasks:

1. Design and implement watchlist functionality.
2. Implement rating and review functionality.

3. Test watchlist and recommendation functionality.

User Story 4: Access Account Settings and Help

As a user, I want to access my account settings and get help when I need it.

Tasks:

1. Design and implement account settings page.
2. Integrate with Amazon's customer support system.
3. Test account settings and help functionality.

User Story 5: Download and Watch Offline

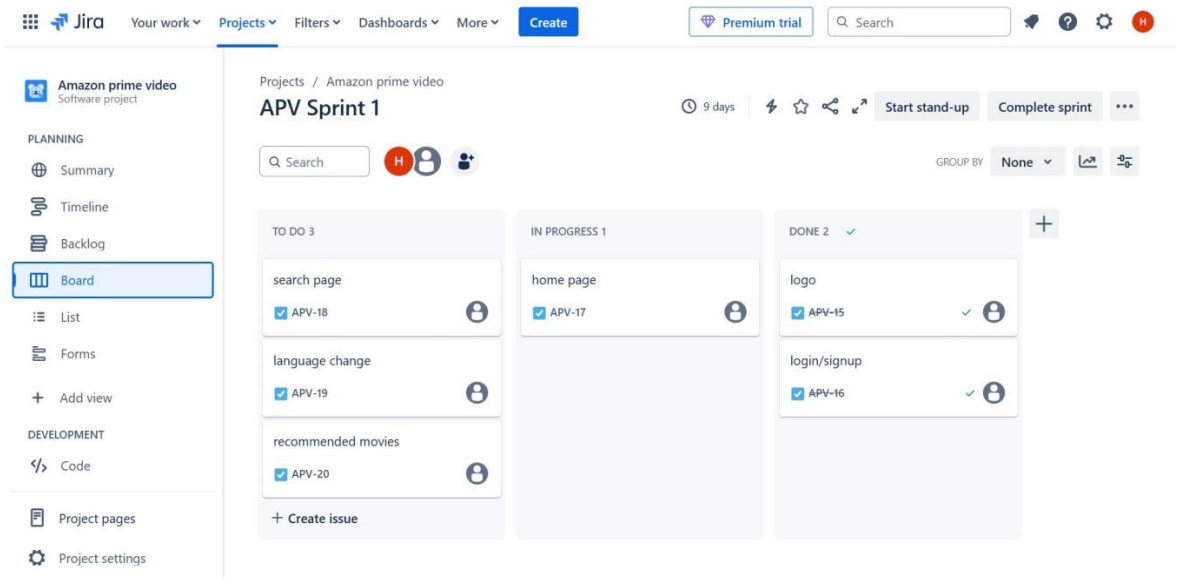
As a user, I want to download movies and TV shows for offline viewing.

Tasks:

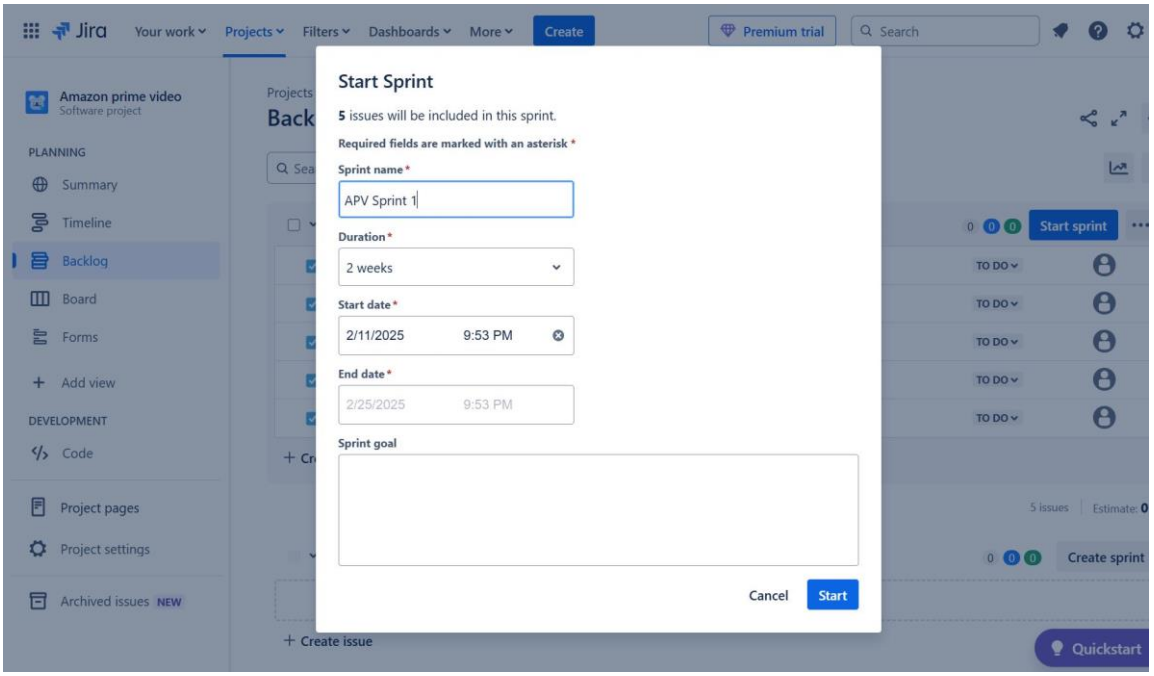
1. Integrate with Amazon's download engine.
2. Design and implement offline viewing functionality.
3. Implement download management functionality.

3. Configure JIRA for the managing the project to solve the identified problem.

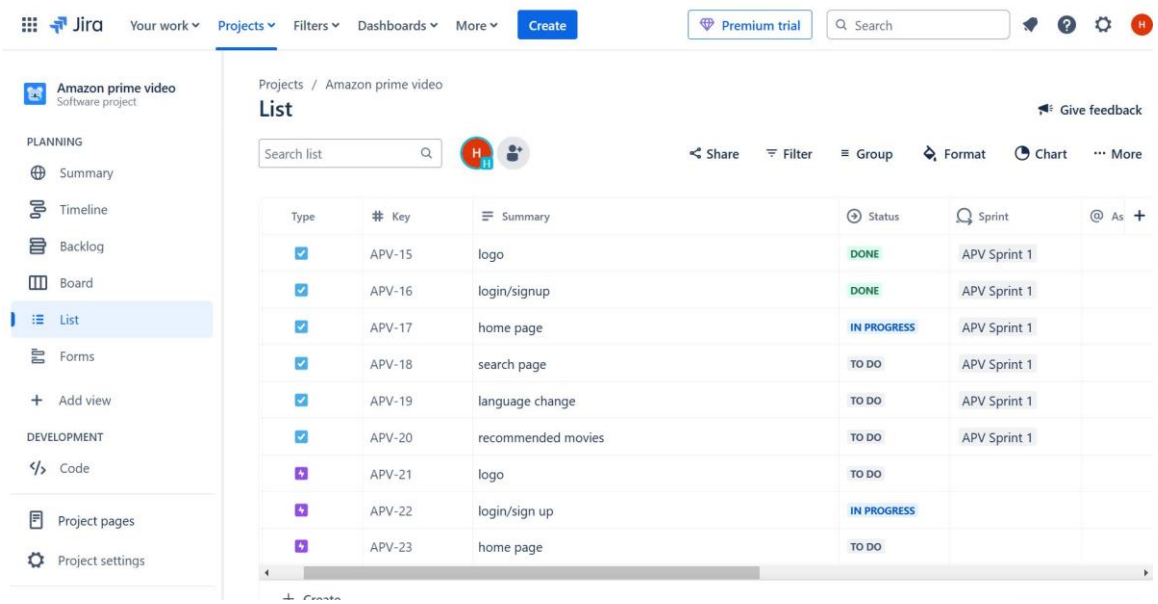
Board:



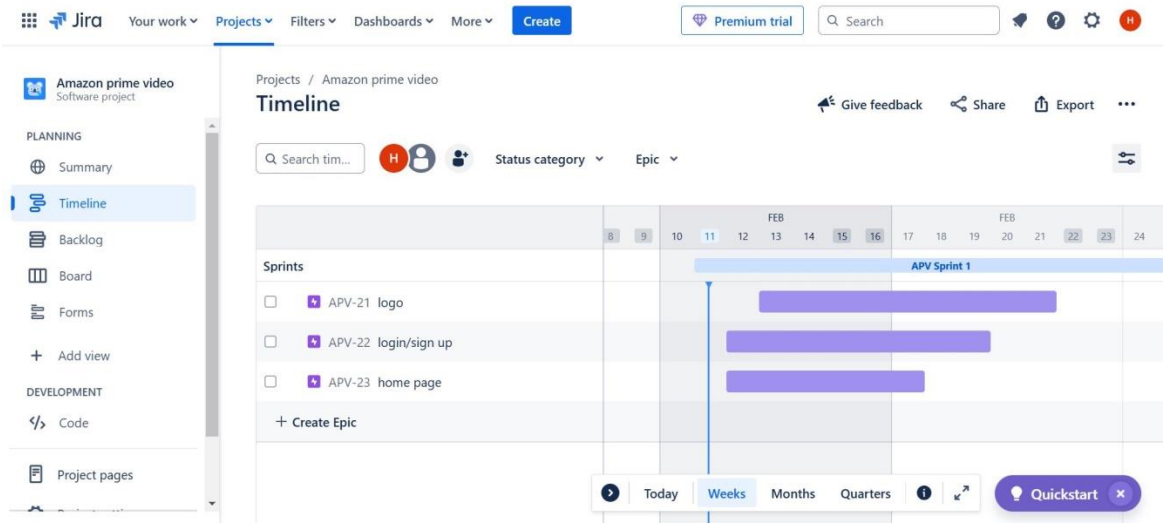
Start Sprint:



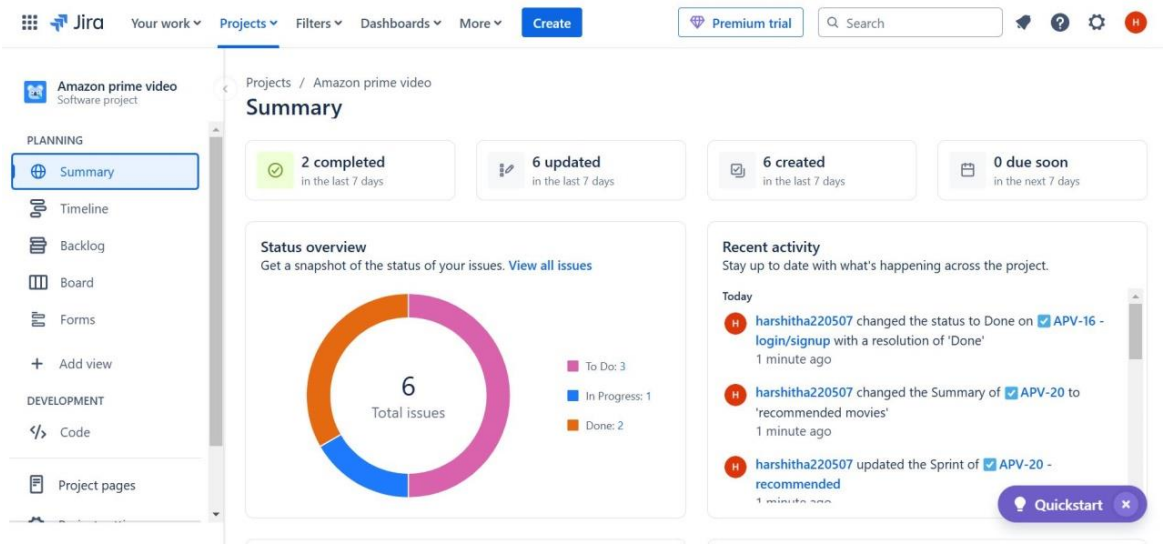
List:



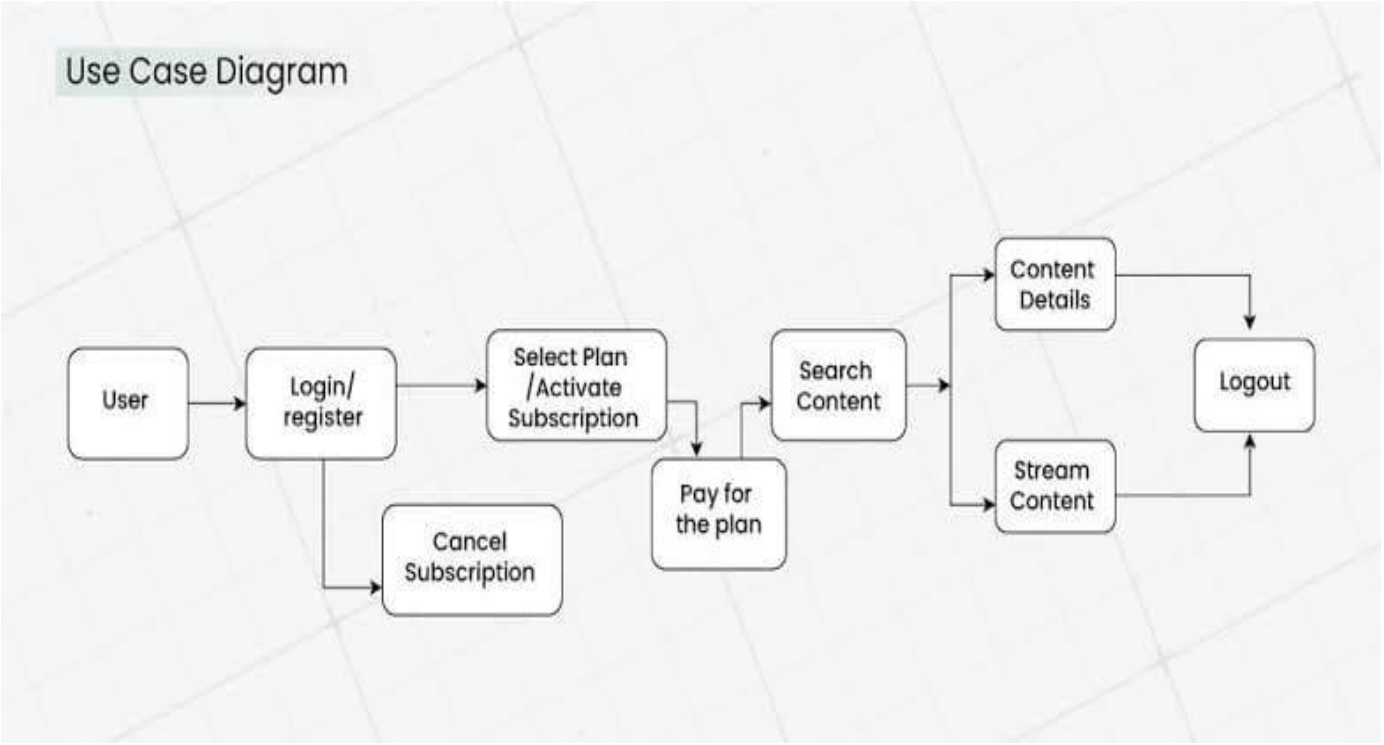
Timeline:



Summary:



4. Draw UML diagram for given case.



WEEK-07

1). Create detailed user stories for the above identified problem.

Epic: Prime Video.

User Story 1: Search and Discover Content
As a user, I want to search for specific movies and TV shows, and discover new content based on my interests.

Acceptance Criteria:
- The user can search for movies and TV shows by title, genre, or actor.
- The user can browse through curated content lists, such as "Trending Now" or "Recommended for You".
- The user can view detailed information about each title, cast, and reviews.

Identified Problem: The search functionality is slow and returns irrelevant results, making it difficult for users to find what they're looking for.

User Story 2: Play and Control Video
As a user, I want to play movies and TV shows, and control the video playback experience.

Acceptance Criteria:
- The user can play movies and TV shows in high definition.
- The user can control video playback using buttons for play, pause, fast forward, and rewind.
- The user can adjust video settings, such as brightness, contrast, and subtitles.

Identified Problem: The video playback is choppy and buffers frequently, disrupting the user's viewing experience.

User Story 3: Manage Watchlist and Recommendations
As a user, I want to manage my watchlist and receive personalized recommendations based on my viewing history and preferences.

Acceptance Criteria:
- The user can add and remove titles from their watchlist.
- The user receives personalized recommendations based on their viewing history and preferences.
- The user can rate and review titles to improve recommendations.

Identified Problem: The recommendations are not accurate and do not take into account the user's viewing history and preferences.

User Story 4: Access Account Settings and Help
As a user, I want to access my account settings and get help when I need it.

Acceptance Criteria:
- The user can access their account settings, including profile information and payment details.
- The user can contact customer support through email, phone, or live chat.
- The user can access a comprehensive help center with FAQs and tutorials.

Identified Problem: The account settings are difficult to navigate and the help center is not comprehensive, making it hard for users to find the help they need.

User Story 5: Download and Watch Offline

As a user, I want to download movies and TV shows for offline viewing.

Acceptance Criteria:

- The user can download titles for offline viewing on their mobile device.
- The user can access downloaded titles through the Prime Video app.
- The user can watch downloaded titles without an internet connection.

Identified Problem: The download process is slow and the offline viewing experience is not seamless, with frequent errors and interruptions.

2). Organize and play planning poker to decide on user points.

When estimating story points using Planning Poker for features in an app like Prime Video, you would typically consider factors like complexity, effort, risk, and dependencies. Here's a suggested estimation for each feature based on a relative scale:

Estimation	User points
User registration/login	2
Plan selection and activation	3
Payment processing	5
Content browsing	8
Search content	13
Streaming content	21
Log out	3

Reasons for the user points and estimation:

1. **User Registration/Login:2 points**

   This task involves user authentication, possibly using third-party logins, secure password management, and error handling.

2. **Plan selection and Activation:3 points**

   This feature involves showing different subscription plans, presenting relevant pricing, activating plans, and managing payment methods.

3. **Payment Processing:5 points**

   Payment integration, such as connecting with payment gateways, ensuring secure transactions, handling user data, and dealing with retries or failures, can be quite complex and has higher risk.

4. **Content Browsing:8 points**

   Content browsing can range from showing a grid of titles to managing categories or genres. It requires UI implementation and data fetching but is generally less complex than payments or plan activation.

5. **Search Content:13 points**

   A search feature with filters, autocomplete, and showing results can be moderately complex, especially if it integrates with a backend system or uses recommendation algorithms.

6. **Streaming Content:21 points**

   Streaming involves handling video playback, dealing with buffering, and implementing a smooth, high-quality experience across multiple devices.

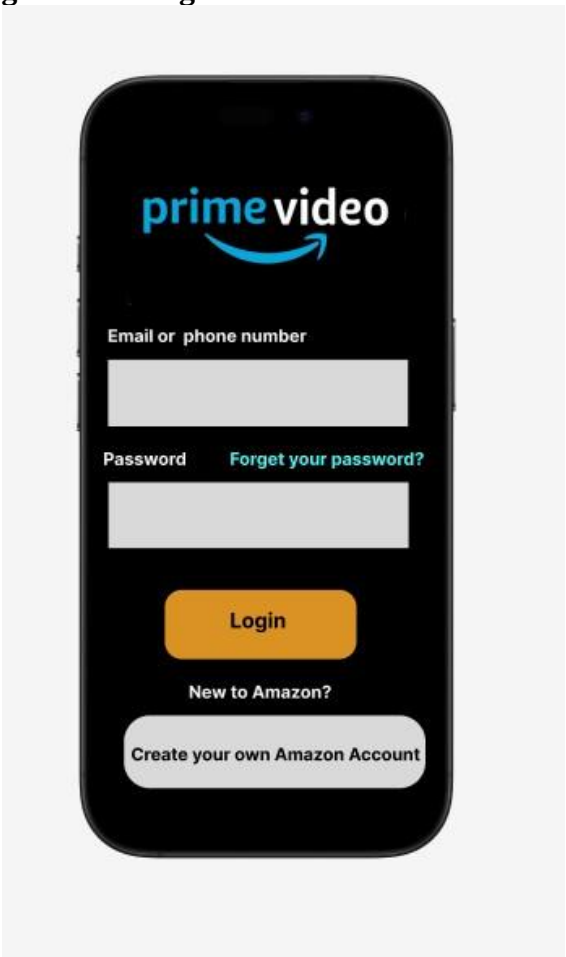
7. **Log Out:3 points**

   This is a relatively simple feature to implement, mainly involving user session termination and cleanup.

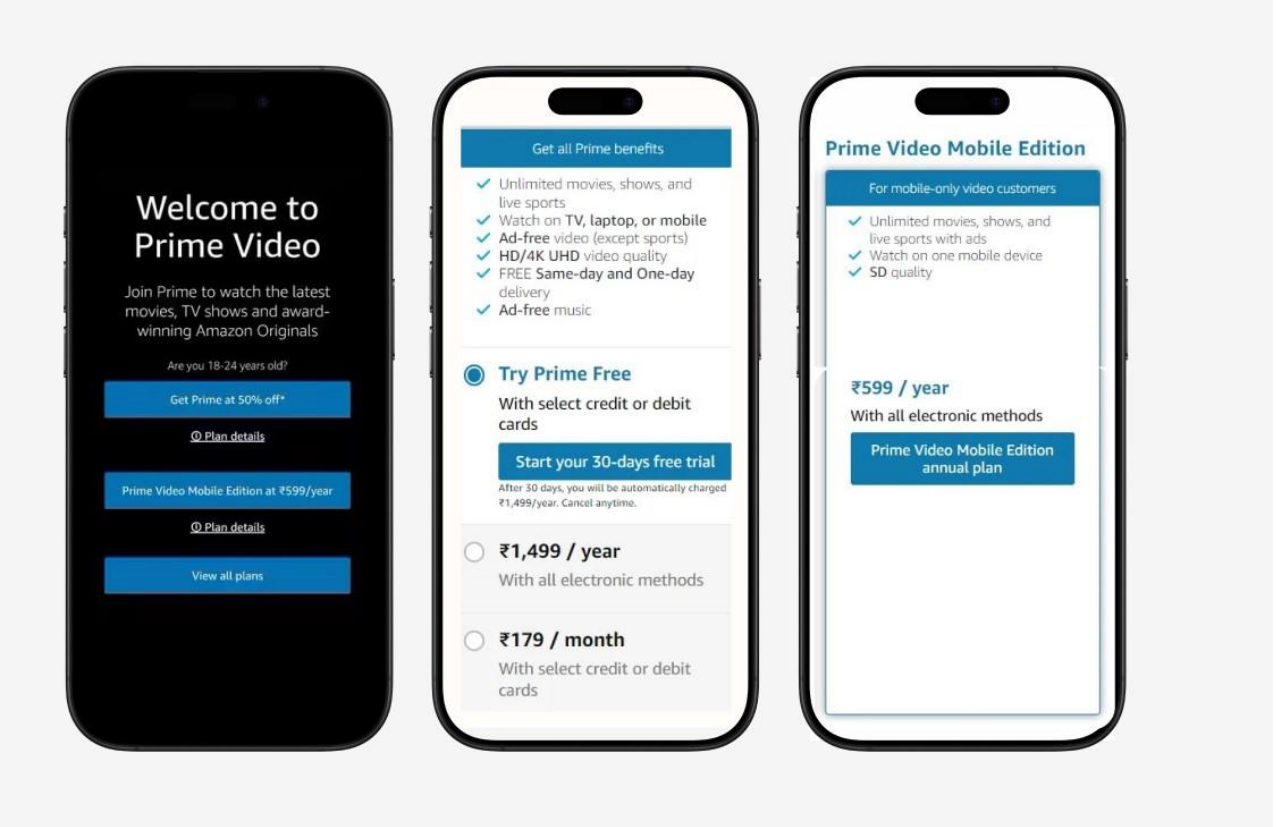
WEEK-08

1) .Create sitemap and wireframe for above created user stories.(Tools such as sketch, Adobe XD, Figma, etc can be used)

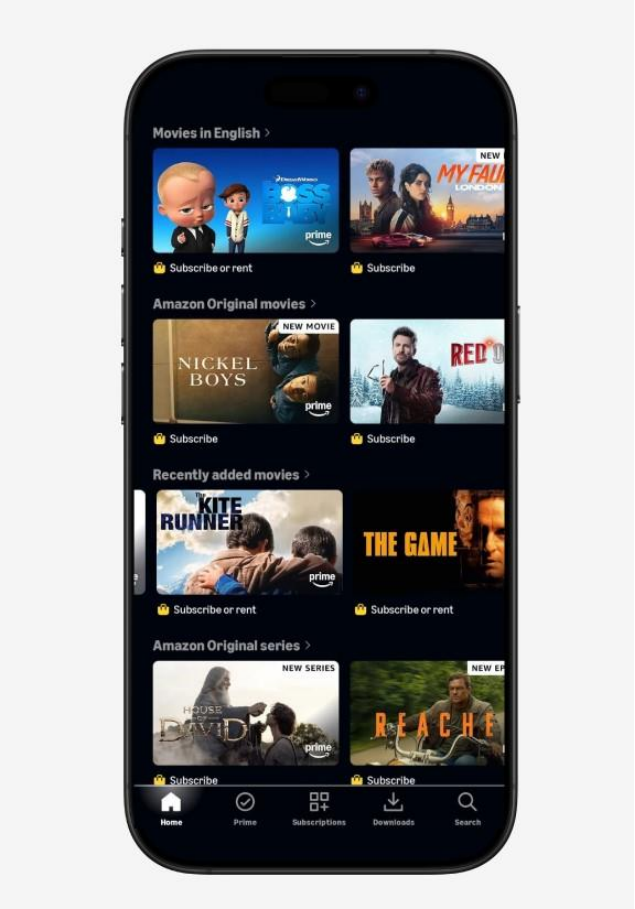
1. User Registration/Login



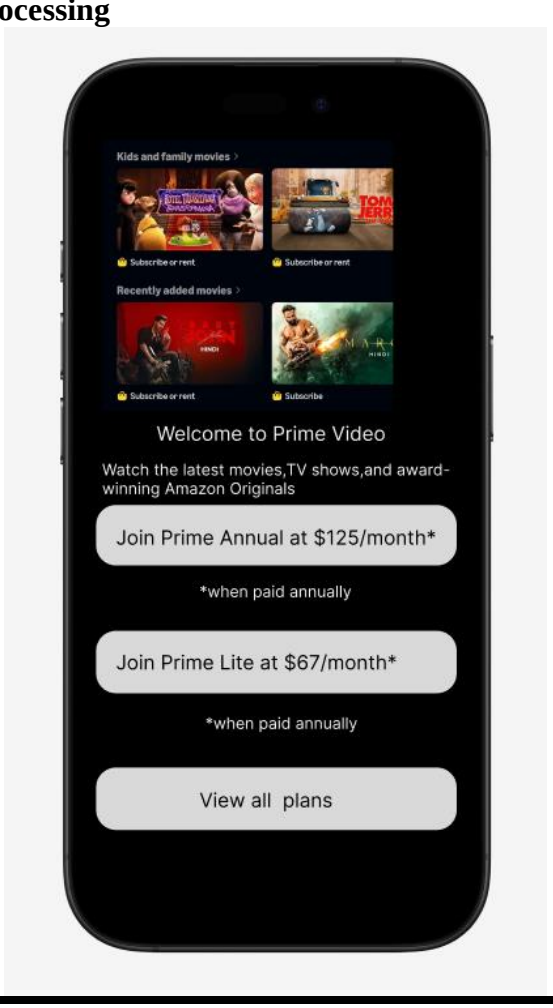
2. Select plan and Activation



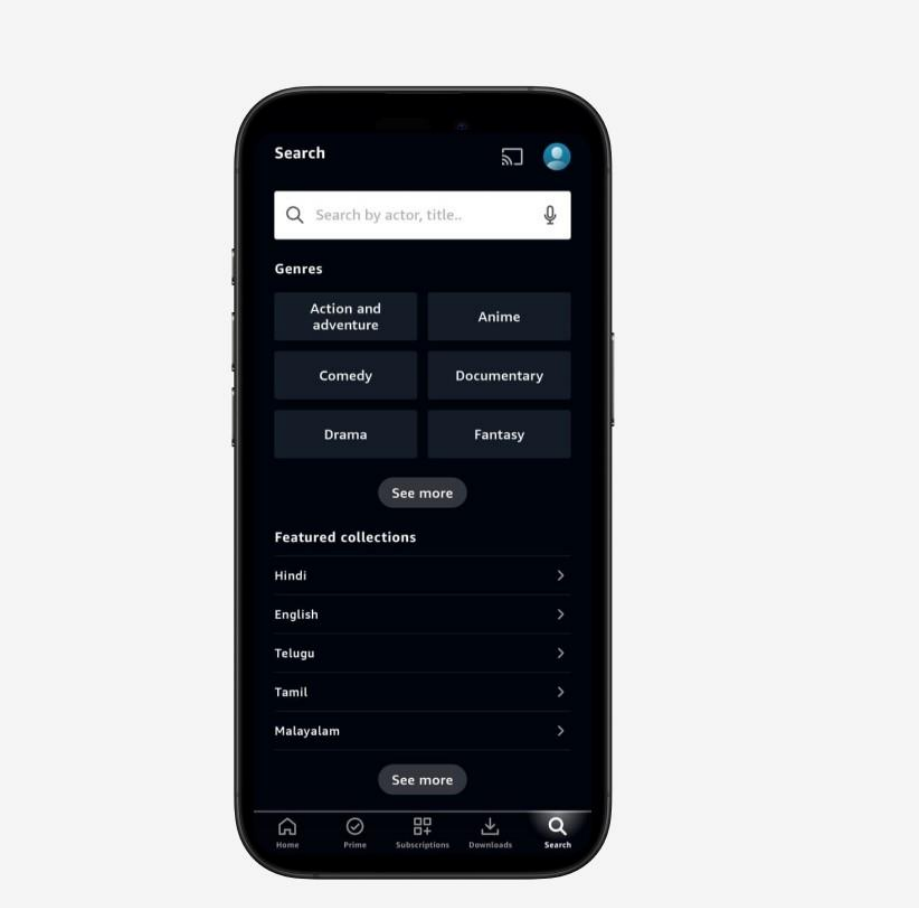
3. Content Browsing



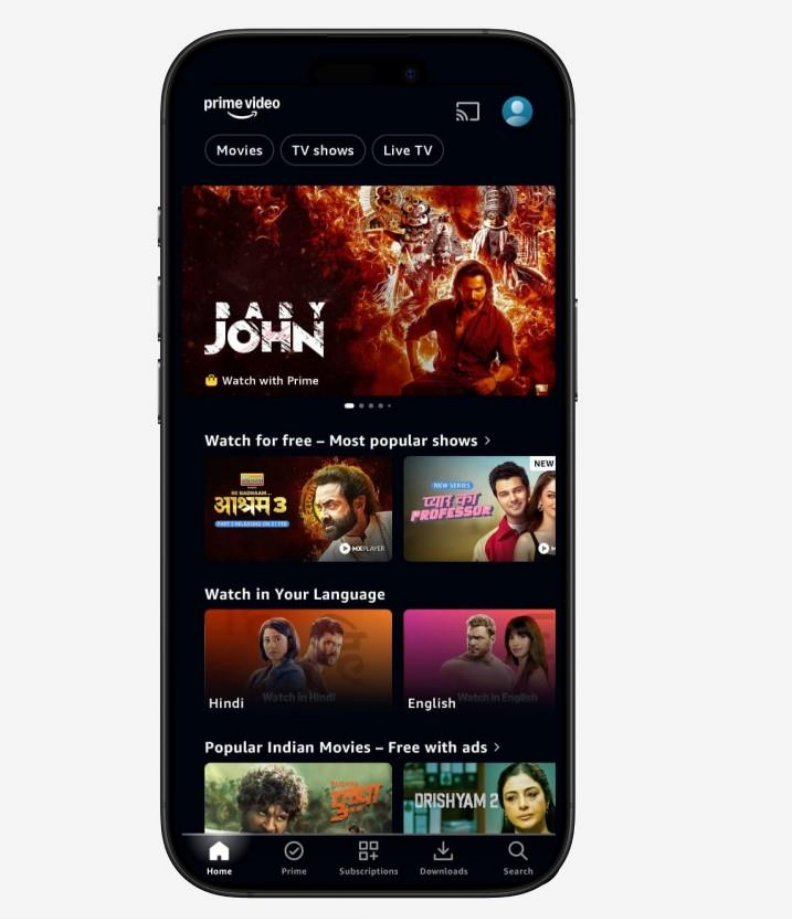
4. Payment Processing



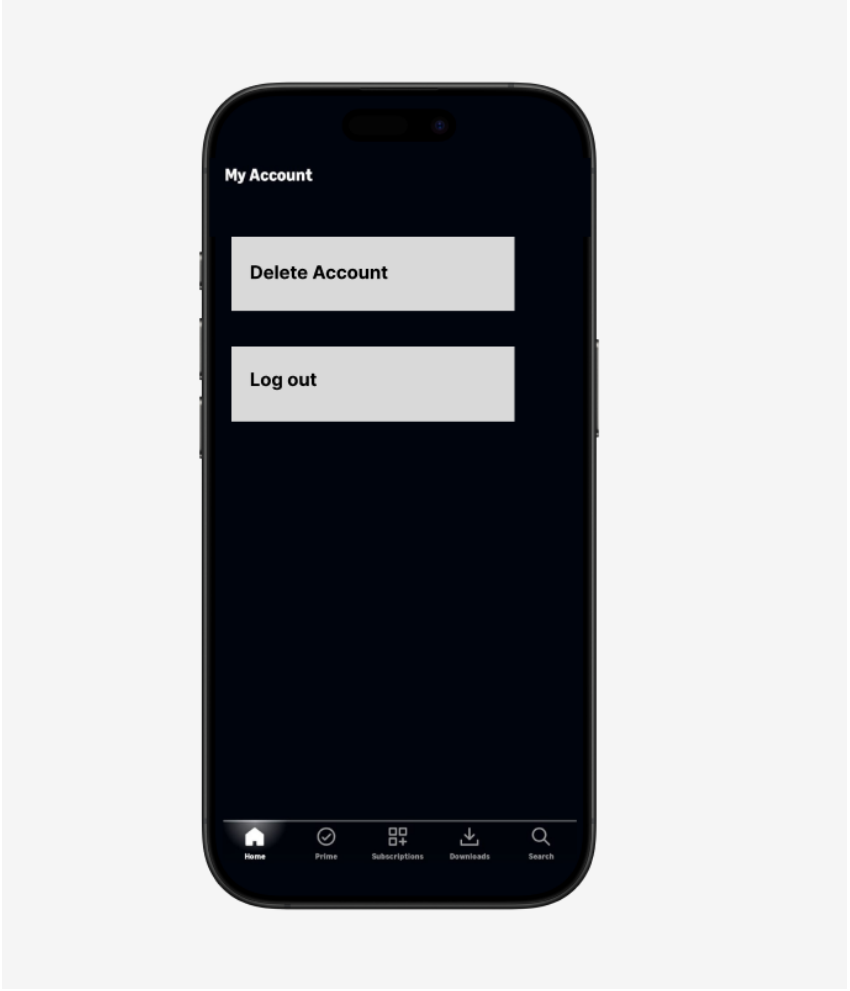
5. Search Content



6. Streaming Content



7. Log out


My Account
Delete Account
Log out
Home
Prime
Subscriptions
Downloads
Search