PlanVerse

Project Scenario (Introduction)

- **Concept:**

  Planverse is a comprehensive project management website that empowers individuals and teams to efficiently organize and collaborate on their projects. Planverse offers a user-friendly interface and a wide range of features to facilitate project management workflows.

  At Planverse, our mission is to provide a versatile platform that answers to both personal and team-based project management needs. Whether you're an individual looking to manage your personal tasks or a team in need for seamless collaboration, Planverse has got you covered.

- **Why Planverse (Problem description):**

  There are several problems that Planverse, as a project management platform, can help solve:

  - Organizational Challenges: Planverse provides a structured environment to organize tasks, projects, and workflows. By providing a centralized location for project-related information, this platform eliminates the need for scattered documents, spreadsheets, and communication channels, making it easier to find and access relevant information.
  - Collaboration and Communication: Planverse facilitates effective collaboration and communication among team members. It provides

features such as shared workspaces, chatrooms, commenting, voice/video calls and real-time updates, reducing the reliance on emails, meetings, and other communication channels, improving efficiency and promoting transparency within the team.

- Agile Project Management: Many project management platforms, including Planverse, are designed to support Agile methodologies such as Scrum or Kanban, enabling teams to implement Agile practices effectively. These platforms facilitate iterative and incremental development, promote collaboration, and encourage adaptive planning.
- Information and Document Management: Planverse offers a centralized repository for storing and managing project-related documents, tasks, milestones and other resources. This eliminates the need for scattered file storage systems and ensures that all project-related information is easily accessible to the team.
- Flexibility and Customization: Planverse offers flexibility and customization options to adapt to different project management approaches and workflows. Users can customize boards, lists, and cards to match their specific requirements and preferences.

Technologies, tools and hardware requirement

Planverse is a web-based application that offers a professional project management solution. With its web-based nature, Planverse eliminates the need for specific hardware capabilities, as it can be accessed on any device equipped with a modern web browser. Whether accessing the platform from a

desktop computer, laptop, tablet, or even a mobile device, users can enjoy the full functionality of Planverse.

Planverse is developed using tools and technologies such as:

- React, Redux, RxJs, Typescript and SCSS for Front-end development
- Golang, Echo and Postgresql for Back-end and database implementation
- RESTFUL api for communication between Back-end and Front-end
- Docker to build, test and deploy

User Scenarios

In the following section some of the main user scenarios are described:

- **Authentication**
  - **User Registration:**
    - The user visits the Planverse website.
    - They navigate to the signup page.
    - They provide their email address and create a password.
    - They need to verify their email address through a confirmation code sent to their inbox.
    - Upon successful registration, the user is redirected to their dashboard page.
  - **User Login:**
    - Users enter their registered email address and password on the login page.
    - They click the login button to submit the login credentials.

- If the provided information is valid, the user is authenticated and granted access to their Planverse account.
- Upon successful login, the user is redirected to their dashboard page.

- **Functional Requirements:**
  - User Registration:
    - Provide a registration form where users can enter their email address and create a password.
    - Implement validation checks for email format and password strength.
    - Store the user's email address and securely hash their password for storage.
    - Implement an email verification process to confirm the user's email address.

  - User Login:
    - Implement a login form where users can enter their registered email address and password.
    - Validate the login credentials against the stored user data.
    - Authenticate the user if the provided information is correct.
    - Implement secure session management to maintain the user's login state.

  - Error Handling:
    - Display appropriate error messages for invalid login attempts, such as incorrect email or password.

        - Handle scenarios where the user forgets their password and provide a password reset mechanism.
        - Remember Me Functionality: an option for users to enable the "Remember Me" feature for automatic login on subsequent visits.
        - Implement secure cookie-based authentication to maintain the user's session.

    - User Account Management:
        - Allow users to edit their account information, such as updating password, name and profile picture.
        - Implement appropriate validation checks and handle any necessary email address verification or password change confirmation.

    - Security Measures:
        - Implement industry-standard security practices, such as encryption of sensitive data, protection against common vulnerabilities (e.g., SQL injection, cross-site scripting), and secure password storage using techniques like hashing and salting.

- **User Dashboard**
    - **Create Project and Project lists:**

        - Users can see a list of their available projects in their dashboard page.
        - Users can create a new project and get its share link to add other users.
        - Users can manage their projects and have an overview of their projects.
        - Users can join a project by entering the invitation link generated by the project's owner.
    - **Functional Requirements:**
        - Project creation:
            - Provide a form where users can set a name, description, image and other properties to customize their projects.
        - Project lists:
            - Provide a section to have an overview of joined projects and enter each project board/workspace by clicking on the projects icon.
            - Have a user-friendly design to ease the access of each project.
        - Invite link navigation:
            - Provide a mechanism where when a project invite link is pasted in the browser search bar the user will be navigated to the projects join page.

- **User Profile/Setting:**
    - **Manage Profile and Visibility:**

        - Users can change their username that is visible to other users.
        - Users can add bio to give some detail about themself for their profile.
        - Users can add an image to their profile.
    - **Manage Setting:**
        - Users can change their email addresses to get the latest updates of their projects.
        - Users can determine which notifications they want to recieve on their email addresses.
        - Users can set the theme of their working space.
        - Users can delete their account.
    - **Functional Requirements:**
        - Changing profile fields:
            - Provide a form where users can change their username, email address, image and other properties to customize their profile
        - Deleting account:
            - Provide a form where users can delete their account. The app must warn them about the consequences of this action.

- **Project Dashboard:**
  - **Add Members to Project:**
    - Admins can add members to projects and grant them admin privileges.
  - **Customize Project Dashboard:**
    - Admins can change the theme of the project dashboard.
    - Admins can change the name of the project.
  - **List Different Type of Tasks:**
    - Members of project can see project tasks classified in 5 different default type:
      - Backlog: project's all tasks
      - To-do: tasks that are assigned to members but not started yet
      - Doing: tasks that members are working on them
      - Code Review: tasks that the team leader should review before putting it in Done part
      - Done: tasks that were completed successfully
    - Only Admins have access to tasks of Done type.
    - Admins can create custom task types and decide whether other members can add tasks to them or if only admins can do so.
    - Admins can delete a task type.
  - **Task Management:**
    - Members can make changes in tasks. These changes contains:
      - Writing comment
      - Changing the theme color of a task card

- Admins can create new tasks or make changes in other tasks. These changes contains:
  - Adding members
  - Adding labels (to indicate the field in which the task belongs)
  - Writing description
  - Writing comment
  - Changing the theme color of a task card
- Admins can change the type of every task, but other members can only change the type of those that don't have limited access.

- **Functional Requirements:**
  - Creating list:
    - Each type of task is a list that contains tasks cards of that type
  - Creating task:
    - Provide a mechanism for admins to create tasks in different lists
  - Adding member:
    - Provide a mechanism for admins to add members to project

- **Members Communication:**
  - **Project Chatroom:**
    - Each project has a text channel for members.
    - Members can send and receive text based messages.
    - Members can upload and download files.
    - Admins can make changes to the text channel. These changes contains:
      - Changing theme
      - Removing member from chatroom
      - Adding member from project to chatroom
  - **Project Voice call:**
    - Each project has a voice channel for members.
    - Members can speak to each other via this environment.
    - Admins can make changes to the voice channel. These changes contains:
      - Changing theme
      - Removing member from voice channel
      - Adding member from project to voice channel
  - **Functional Requirements:**
    - Provide an environment for each project's members to communicate with each other via texting
    - Provide an environment for each project's members to communicate with each other via speaking

Non Functional Requirements:

- Privacy & Security:
    - PlanVerse should prioritize privacy and security of users data
    - Using strong encryption for saving data like password or OTP in database
    - Using JWT for authorization
    - Using TLS for transferring data between clients and server
- Performance:
    - PlanVerse should provide fast responses for users
    - Using redis along with postgres to increase performance
    - Using middlewares like caching to increase performance
- Maintainability:
    - PlanVerse should be maintainable and upgradeable
    - Using clean code to let the application be maintainable
    - Writing a modular code
- Availability:
    - PlanVerse should provide reliable access for users.
    - Using backup system will minimize the server downtime

Requirements Validation

- **Invite link navigation:**
    - This requirement is clear and concise. This requirement is not against our system constraints and it's also testable. It can be tested easily by sending an invite link and seeing if another user gets the link or not.
- **Project creation:**
    - This requirement is clear and concise. However, it would be helpful to specify what other properties are required to customize their projects. The requirement doesn't violate any system constraints. The requirement can be tested by a user entering project properties and checking if the project is created or not.

RCA(Root Cause Analysis)

- **Issue: Delay in Notification Delivery**
    - Potential Root Causes:
        - Server Performance Issues: The app's server may be overloaded, causing a delay in processing requests, including sending out notifications.
        - Third-Party Integration Lag: Integrations with email or other messaging services could be experiencing lag or downtime, delaying notification dispatch.

        - App-side Queue System Inefficiencies: The app's internal queue system for notifications may not prioritize notifications efficiently, causing backlogs and delays.
    - RCA Process:
        - Analyze server load reports to identify performance spikes or bandwidth issues.
        - Check service status and incident logs of third-party integration partners.
        - Review the app's notification queuing logic and processing time metrics for potential optimizations.
- **Issue: Frequent App Crashes on User's Phones**
    - Potential Root Causes:
        - Incompatibility with Multiple Phone Models: The app may not be properly optimized for all phone hardware, causing crashes on unsupported or less common models.
        - Memory Leak in the New Update: A recent update might have introduced a memory leak, causing the app to consume excessive resources and crash.
        - Corrupted App Data or Cache: Local corruption of app data or cache could lead to unexpected behavior and crashes.
    - RCA Process:
        - Collect and analyze crash reports, paying attention to device models and operating systems.
        - Perform a version rollout analysis to correlate the onset of crashes with recent app updates.

        - Implement remote logging to detect patterns of corrupted data leading to crashes.
- **Issue: Slow Task Synchronization Across Devices**
    - Potential Root Causes:
        - Network Latency Issues: Poor server response times can lead to slow synchronization of tasks across users' devices.
        - Database Bottlenecks: The app's backend database may be inefficiently structured, resulting in slow read/write operations during sync.
        - Concurrent Access Locking: High levels of concurrent user access may cause locking in the database, leading to delayed task updates or sync operations.
    - RCA Process:
        - Conduct ping tests and trace routes to identify any network latency.
        - Analyze database performance and indexing to find slow query executions.
        - Investigate database transactions and locking mechanisms during peak usage times for possible optimizations.
- **Issue: Inaccurate Project Progress Tracking**
    - Potential Root Causes:
        - User Input Errors: Inconsistent or incorrect data entry by users can lead to inaccurate tracking of project progress.

        - Unsupported Multi-user Editing: Lack of real-time updates when multiple users are editing a project simultaneously may result in conflicting information.
        - System Time Zone Differences: The app may not be handling time zone differences correctly, leading to skewed timelines and progress tracking discrepancies.
    - RCA Process:
        - Review the user interface and data entry points to identify opportunities to reduce user error, such as implementing input validation or clearer instructions.
        - Assess the app's handling of concurrent edits and implement real-time collaboration features or conflict resolution protocols.
        - Test time zone functionalities and check system clock synchronization for globally distributed teams.
- **Issue: Difficulty in Customization of Workflow Setups**
    - Potential Root Causes:
        - Limited Feature Exposure: Users might not be aware of all the customizable features due to inadequate in-app guidance or documentation.
        - Complex User Interface: The complexity of the user interface may hinder users' ability to customize their workflow effectively.
        - Insufficient API Capabilities: The app's API may lack features that advanced users require for deep customization or automation.
    - RCA Process:

- Investigate user engagement with help content and update guides, tooltips, and tutorials to promote customization features.
- Perform usability testing to identify which aspects of the UI complicate workflow customization and streamline the user experience accordingly.
- Review the API's functionality and consider enhancing it based on user feedback and usage patterns.

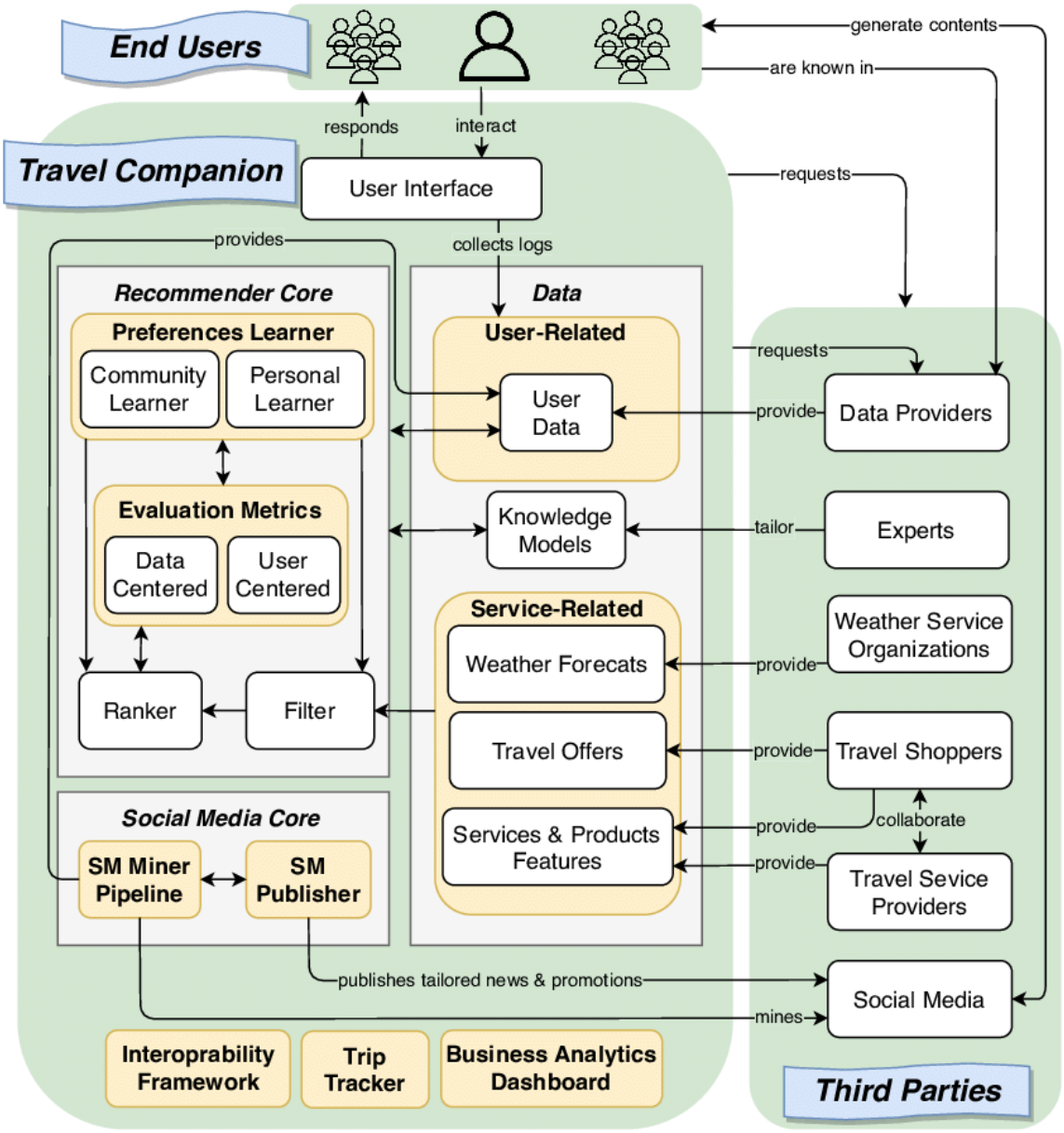
Conceptual Architecture

The project manager system consists of three main components: the user interface, the application server, and the database. The user interface is responsible for providing an intuitive and user-friendly interface for users to interact with the system. The application server is responsible for processing user requests and generating responses. The database is responsible for storing and retrieving data related to the project manager system.

- The user interface component consists of several sub-components, including the dashboard, the project workspace, and the task list. The dashboard provides users with an overview of their projects and tasks. The project workspace allows users to create and manage projects. The task list allows users to create and manage tasks associated with their projects.
- The application server component consists of several sub-components, including the API gateway, the authentication service, and the business logic layer. The API gateway is responsible for routing requests from the user interface to the appropriate service.

The authentication service is responsible for authenticating users and authorizing access to the system. The business logic layer is responsible for processing user requests and generating responses.

- The database component consists of several sub-components, including the project database, the task database, and the user database. The project database stores information related to projects, such as project name, description, and status. The task database stores information related to tasks, such as task name, description, and status. The user database stores information related to users, such as user name, email address, and password.


End Users
generate contents
are known in
responds
interact
Travel Companion
User Interface
requests
provides
collects logs
Recommender Core
Preferences Learner
Community Learner
Personal Learner
Evaluation Metrics
Data Centered
User Centered
Ranker
Filter
Data
User-Related
User Data
Knowledge Models
Service-Related
Weather Forecats
Travel Offers
Services & Products Features
Social Media Core
SM Miner Pipeline
SM Publisher
requests
provide
Data Providers
tailor
Experts
Weather Service Organizations
provide
provide
Travel Shoppers
collaborate
provide
provide
Travel Sevice Providers
publishes tailored news & promotions
mines
Social Media
Interoprability Framework
Trip Tracker
Business Analytics Dashboard
Third Parties

Stakeholders

In the context of Planverse, the stakeholders can be identified as follows:

- Users: The primary stakeholders of Planverse are the users themselves. They include individuals, teams, and organizations who utilize the platform for project management purposes.

- Development Team: The development team, including software engineers, designers, and product managers, are key stakeholders involved in the creation and maintenance of the Planverse platform. They are responsible for designing, developing, and continuously improving the features and functionalities based on user feedback and industry best practices.

- Management and Executives: The management and executives within the organization that owns and operates Planverse are stakeholders with a strategic interest in the platform's success. They oversee the overall direction, business goals, and financial aspects of Planverse, ensuring its alignment with the organization's objectives.

- Project Managers: Project managers, whether they are part of the development team or external users, are stakeholders who rely on Planverse as a tool to manage and track the progress of their projects.

- Partners and Integrators: Partners and integrators who collaborate with Planverse, such as third-party service providers, API integrators, or consultants, are stakeholders with an interest in the platform's compatibility.

W5H2

- ## First scenario: Authentication
    - What:
        - Register or logging in to PlanVerse
        - Verify user’s email by sending an OTP
    - Who:
        - Individuals who want to manage their projects
        - Developers who implemented PlanVerse
    - When:
        - By the time we want to start using PlanVerse
    - Where:
        - The sign up/sign in web page
    - Why:
        - Each user needs an account to save it’s information irrespective to other users
        - PlanVerse should authenticate and authorize users to provide them different accesses
    - How:
        - Register: user sends its credentials. Then the user will receive a verification email. If the user’s email is verified successfully, its

information will be saved in the database and the user registration will be completed
        - Login: user sends its credentials. If the credentials sent by user and database information are the same, then the user will be logged in
    - How Much:
        - Register: user credentials are username, password, email address and password repeated
        - Login: user credentials are username and password

- **Second Scenario: Defining Project**
    - What:
        - Creating a project
        - Adding members to the project
        - Determining lists (tasks types)
        - Determining each list's tasks
    - Who:
        - Individuals who want to manage a project.
        - Developers who implemented PlanVerse
    - When:
        - Any time a user wants to create a new project
    - Where:
        - At user dashboard, each user can decide to make a new project or work on an existed project
    - Why:
        - Each user should be able to create a team to work on a project

        - Each user should be able to create a work space to work on their projects
    - How:
        - Users will choose a name for the project and create a project. Then they will add members to the project or they could work alone on it. Then users will create types of tasks in the project dashboard and determine the tasks in each type

    - How Much:
        - The Owner of project should determine project name, project lists (type of tasks), project tasks, project members and some other details about project

GSM

- **Reliable Notification System**
    - Signal: users get notifications through email or as push notification based on their setting for changes in their projects
    - Metrics: compare the number of change requests and notifications sent to make sure we provide a reliable notification system
- **Good performance**
    - Signal: users feel no delay in their activity as the request and responses are delivered seamlessly
    - Metrics: measure average request and response time and users ratings

- **Safe and secure data management**
    - Signal: users can undisturbedly use their passwords and communicate with each other as the sensitive data and messages are encrypted and safe from any abuse and breach
    - Metrics: Increase in number of users and messages as it demonstrates that PlanVerse has become a reliable platform for users to have a safe and secure experience