Planify: Learning Platform for Creating and Sharing Roadmaps

Salahaldeen Tanbour
Mohammad Ahmad

Supervisor: Dr. Alaadin Masri

June 27, 2025

Contents

List of Figures

Abstract

Planify is a platform for creating and sharing roadmaps made by users for users. Roadmaps are an essential part of any learning process; Whether learning a skill or a topic, students should follow well-made and reviewed roadmaps to act as a guide.

It features the creation of user roadmaps, the ability to share those roadmaps on a platform that other learners can explore and possibly enroll. With the ability to track progress and increase user-retention and performance.

Planify profits by serving isolated spaces for companies and organization to share their proprietary and internal resources privately and manage their own users with the system.

Built on a monolithic architecture with ASP.NET Core for the back-end, Python for the recommendation engine, and both React & React Native for the front-end.

Acknowledgment

We would like to express our gratitude to everyone who helped making this project possible. Special thanks to our supervisor: Dr. Alaadin Masri for his support and guidance throughout the project period.

Chapter 1

Introduction

1.1 Context

Roadmaps are an essential part of any learning process; whether learning a skill or a topic, students should follow well-made and well-reviewed roadmaps to act as a guide. It is easy to wander off your path when you are not going in a specific direction. Roadmaps can solve this issue by listing topics and concepts in a top-down approach, where you can keep track of the bigger picture and where your skills lie.

All other learning methods like courses, books, mentorship, they all follow a line of topics or an order to follow so that info gets gradually without skipping over important topics, all those media types are made in such a way the student gets to a topic when they're ready to do so.

And now, the internet has many free learning resources that ranges for nearly every field, but a problem with

1.2 The Gap

With that being said, what we are looking for is basically a structure for learning, telling us the order of things we need to learn. So there needs to be a platform where we can find those directions of any topic we want, and self-learn those skills or get a general idea about a topic without investing deep into a course or book without seeing the big picture.

Moreover, some organizations may have their own learning content internal to their employees and need a way to manage and tailor these experiences to their specific internal needs.

The importance of such a platform becomes more evident when we consider institutions, companies, or learning communities that want to create private, branded, and self-managed learning environments for their members. A one-size-fits-all platform does not scale in different learning contexts.

1.3 Proposed Solution

Here is where Planify comes in; it is a platform that allows users to build and design their own roadmaps, and share them with other users in the system, where those can enroll in them and self-track their progress to self-assess their improvement and completion.

Planify features a builder to design and structure a graph to represent the topics and content of a roadmap. It's highly customizable to fit most needs. A markdown text editor to display the content beautifully.

In addition to the public platform, we propose a multi-tenant nature that enables each organization to have an isolated and customizable environment for building, sharing, and managing structured learning roadmaps.

Planify can track many organizations by isolating data from each organization into its own infrastructure, so that data from two different organizations do not get mixed. This multi-tenant behavior allows companies to post proprietary content and share them across their own users in the system. In a way that does not get bloated with content on the public platform.

1.3.1 Features

- Drag and Drop builder with high customization
- Recommendation page with recommendation engine
- Tracking progress for enrollments
- Multi-tenancy for organizations
- Bulk register many users using a csv file
- Use of client settings to customize visible feature set
- Real-time chatting capability
- Roadmap generation using AI
- It features an explore page where users get recommended

1.4 Terminology

To ensure clarity throughout this report, key terms and concepts are defined below:

1.4.1 Concepts

- **Isolated Multi-tenancy**: A software architecture where each tenant (organization) has its own logically separated data and configuration, ensuring privacy and customization.
- **Client Side Rendering (CSR)**: A rendering technique where content is generated in the browser using JavaScript after the initial HTML is loaded, improving interactivity.
- **Infrastructure as Code (IaC)**: The practice of managing and provisioning infrastructure through code rather than manual processes, enabling version control and automation.
- **Role Based Authorization (RBA)**: A security model where access to resources is granted based on a user's assigned role within the system.

1.4.2 Technology Tools

- **Docker**: A containerization tool that packages software and its dependencies into isolated units for consistent deployment across environments.
- **SSL certificate**: A digital certificate that encrypts data between the client and server, ensuring secure HTTPS communication.
- **VPS**: A Virtual Private Server used to deploy and host backend services with greater control and flexibility than shared hosting.
- **CI/CD pipeline**: A set of automated processes that handle integration, testing, and deployment of code to ensure rapid and reliable software delivery.

1.5 Objectives

The primary objectives of this work are:

- **To serve a platform for community-based learning** and target private organizations with their internal resources.
- **Provide an easy and intuitive way to build and design roadmaps**.
- **Provide easy way to track progress** for users enrolling in a roadmap.

1.6 Summary

Planify addresses important gaps in the learning landscape by offering a platform that serves many user-created roadmaps designed to help and benefit as many learners as possible. While allowing users to easily self-track their progress and goals. including a way to sustain itself financially by providing isolated spaces for companies and organizations to post their internal content and manage their users.

Chapter 2

Literature Review

Learning roadmaps represent structured sequences of topics designed to guide learners from foundational knowledge to advanced mastery. These roadmaps resemble dynamic, learner-driven curricula and are gaining traction with the rise of online and self-paced education. There is growing interest in tools that allow learners to visualize, personalize, and share learning journeys.

2.1 Instructional Design and Sequencing

Instructional design frameworks such as the ADDIE model and Dick and Carey model emphasize structured content sequencing and scaffolding—progressing from basic to advanced knowledge in alignment with learning goals [1]. This foundation supports the logic behind learning roadmaps, which aim to reduce cognitive load and enhance retention. The concept of instructional scaffolding also aligns with the idea of community-driven roadmaps, where expert users guide beginners with structured topic flows [2]. Research by Valle Torre et al. [3] further confirms that the order in which content is presented significantly influences learning outcomes.

2.2 Learning Paths in Digital Environments

De Smet et al. [4] explored learning path implementations in learning management systems, concluding that structured, visual learning paths improve user engagement and self-regulation. Their work validates the importance of giving learners a clear sense of direction—mirroring what roadmap-based platforms aim to offer. Additionally, Ovtšarenko [5] reviewed adaptive learning systems and highlighted the growing trend of personalized learning flows. Although our system does not implement adaptive algorithms, it supports personalization by enabling users to create and follow tailored learning sequences.

2.3 Positioning of the Proposed System

While the platform discussed in this work is commercially driven, it draws on educational principles highlighted in the literature. By allowing users to build and share structured roadmaps, it promotes instructional clarity, learner autonomy, and community-based scaffolding. These features are consistent with current research advocating for learner-centered, sequenced, and shareable digital learning experiences.

Chapter 3

Methodology

3.1 Planify (Road map Builder)

3.1.1 Initial Design

We started with planning out pages for the web application, drawing simple wireframes to get an idea of the task at hand and materialize requirements. Link here shows what we started out as a wireframe which was later turned into the final version. `https://excalidraw.com/#json=3NAjijCk2GdNyejrnpLQR,lnYqjXCv45B9VwZBOBgHyw`

The Planify is a web application that allows users to create road maps using a visual interface. This interface consists of a canvas where users can drag and drop nodes to create a flowchart-like structure. Each node represents a step in the road map, and the connections between nodes, you can also upload images into the road map for more information and branding.

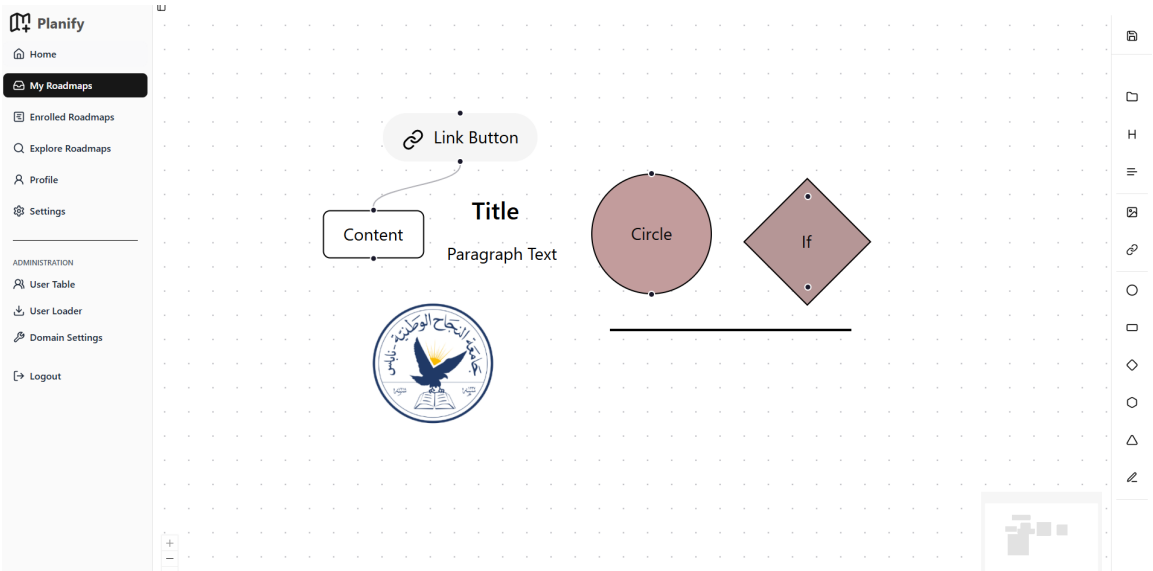

Figure 3.1: Road map nodes

The design shows the following components:

- **Content Node**: The content node is the core component that contains content from the road map creator when the reader presses on it.

- **Link Node**: The link node/button is a special node type that only contains a URL when the reader presses on it, it opens a new tab with the prespecified URL.
- **Title Node**: Title in bold font.
- **Text Node**: Text in regular font.
- **Image Node**: Upload an image from the road map owner device to the road map, the user will be able to view the image and can not change it.
- **Circle Node**: Circle-shaped node without content this type of nodes will be used for building chart flow charts.
- **Diamond Node**: Diamond-shaped node without content; this type of nodes will be used for building chart flow.
- **Line Node**: Static Node with Horizontal line the user can change the length and the width from the node setting tab, this node can be used for decoration.
- **Connection Link**: You can establish connections between different nodes so that the reader will know the next connected-node.

3.1.2 Overview

For a general overview of the platform, we will start by talking about the entities and how relationship works between them. We will start with Users and Roadmaps. a user in the system can have his own created roadmaps, and can have enrolled roadmaps (which are created by other users in the platform)

A roadmap can have many nodes representing its content. It can have many types of nodes as mentioned, and the set of nodes make up the roadmap entity.

When a user enrolls in a roadmap, he has an enrollment that keeps track of his progress. Progress is whether the user has finished the nodes in the roadmap or if they are still in progress. Based on those progresses, we calculate the performance and compare them against other enrollers in the roadmap.

3.1.3 Design Decisions

- We chose a Isolated multi-tenant database architecture to ensure that, when we add a customer, they each get their own database and user accounts with their own application-wide settings. This approach also delivers better performance, scalability, and extensibility: each customer can maintain their own branch of the application with their own developers, and sync bi-weekly or monthly to stay up-to-date with any uploaded patches.
- Using a subdomain-based, isolated multi-tenant approach ensures that users automatically connect to the correct customer instance. This provides a user-friendly, seamless experience by eliminating the need to select which customer to log into at the start.

3.2 System Design and Architecture

3.2.1 Overview of the System

The following figure shows an overview of the system architecture of Planify.

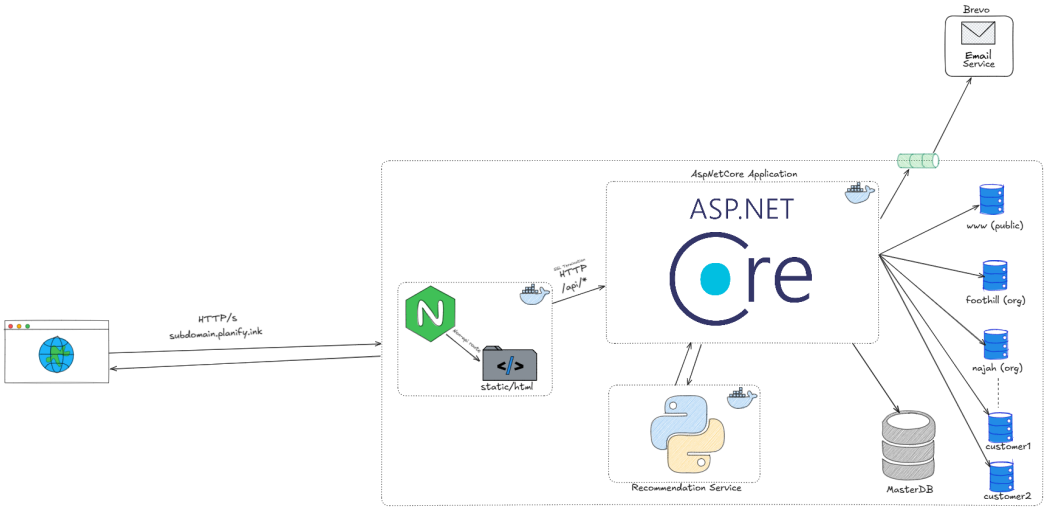

Figure 3.2: Architecture of Planify

The project was developed using a microservices architecture, with separate repositories for each microservice and the frontend client. The repositories are as follows:

- **NGINX**: Works as a reverse proxy to take in all incoming requests and make sure they use HTTPs for secure communication. It routes non-API routes to static/HTML folder to serve deployed React Single Page Application, and routes start with /api/* routed to AspNetCore App which contains the business logic.
- **AspNetCore App**: The main API project for the Planify application. Responsible for authentication and orchestrating workflows traversal & state-management.
- **Recommendation Service**: Recommendation service responsible of recommend road maps to other users based on users skills & interest.
- **Master Database**: Responsible of storing customer related data such as: database connection server, subdomain name.
- **Customer databases**: Each database belong to a specific customer with it's own data, users and customer settings with the application itself.
- **Brevo Email Service**: 3rd party service used to send emails/notification to user's email.

3.2.2 Request/Response Life Cycle

To put all of these moving elements together and to further explain 3.2, we will go over the request life cycle starting from the client and getting back the response to the client.

1. First, the client sends a request, and it must be following HTTPs protocol for secure connection, if not, NGINX is configured to redirect all HTTP requests to HTTPs.

2. After that, when request reaches NGINX, it routes it according to what is stored in the configuration file, it routes requests with route /api/** to the Planify backend docker container.

3. This request will have the tenant name in the sub-domain (eg. najah.planify.ink/), this tenant will be used by planify to first fetch the database connection string from the shared master database. This connection string is crucial to be able to access the tenant database.

4. Then connection to isolated database is made and the request is handled and response is returned to client.

3.3 Backend

3.3.1 Overview

The backend of Planify uses a multi-tenant architecture for databases for scalability and modularity. It consists of:

- **AspNetCore Application**: Handles authentication, road maps, chatting, and business logic using .NET 9 AspNetCore.
- **Recommendation Service**: Built using Python3 with FastAPI Web framework, for NLP we use spaCy with pre-trained English medium-sized model **(in core web md)**.
- **Databases**: MariaDB used for both **customers databases** and **MasterDB**.

3.3.2 Database Diagram

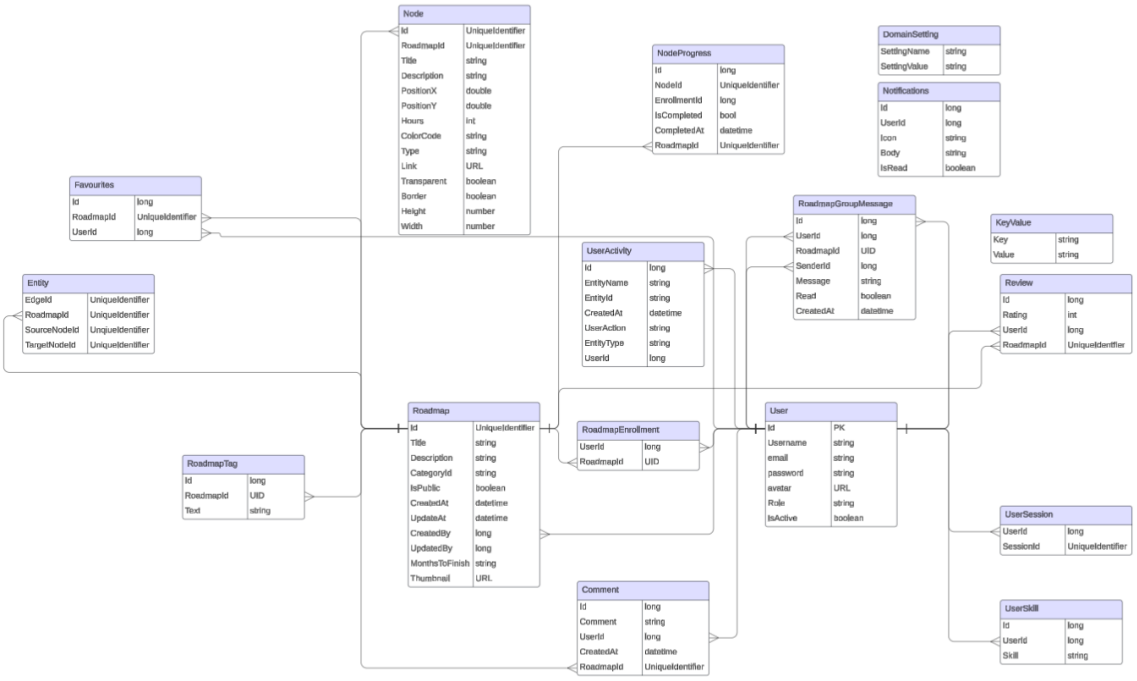

Figure 3.3: Road map nodes

3.4 Frontend

3.4.1 Overview and Architecture

The frontend of Planify is built with **React Native**, providing a cross-platform mobile development solution and **React** for single page application that uses **CSR**. The application follows a component-based architecture with clear separation of concerns:

- **Presentation Layer**: React Native components and screens
- **State Management**: ReactContext with middleware for side effects
- **Service Layer**: API client and Long polling handlers
- **Utils**: Helper functions and shared utilities

3.4.2 Cross-Platform Design

The application ensures consistent behavior across platforms through:

- **Responsive Design**:
  - Platform-specific UI adjustments
  - Adaptive layouts for different screen sizes

3.5 Infrastructure

3.5.1 Architectural Approach

We used a private server **(VPS)** from scratch and set up the server with the public **IPv4** to host the services we have on it, the good thing we moved within a framework by utilizing containers and docker our services have a high deployability percentage so it wouldn't be that hard to go with cloud providers like **AWS** and **Azure**, or even switch to any infrastructure.

3.5.2 DNS & SSLs

Multi-tenancy requires more than regular **SSL Certificate** and a DNS Record, in order to have the ability to keep your HTTP traffic secure within different domains (subdomain1.site.net, subdomain2.site.net .. e.g) you need a wildcard SSL Certificate connected with your HTTP Server **(NGINX)** in order to have secure connections within different subdomains, we chose SSLs ti get the certificate from, and for the DNS Records we used Namecheap.

3.5.3 CI/CD Pipeline

The CI/CD pipeline is managed with GitHub Actions and follows GitOps principles. Key workflows:

- **Frontend**: Repository the flow runs on code changes. 3.4
  - Build React app.
  - Open a FTPs Connection with the **VPS**.
  - Copy the code within **html/static** in **NGINX**.
- **Backend**: Repository the flow runs on code changes. 3.5 3.6
  - Build an image of the app.
  - Host the image on **DockerHub**.
  - Connection with the server download the image and execute it.

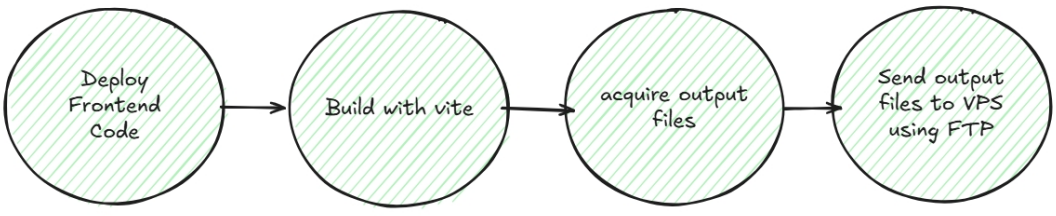

Figure 3.4: Frontend deployment process

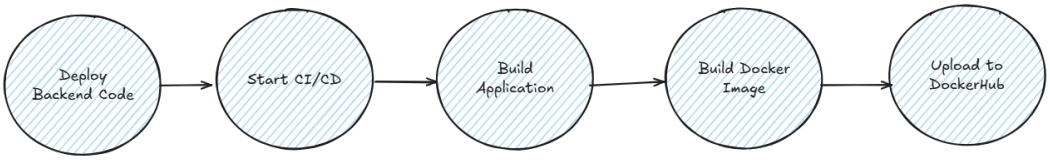

Figure 3.5: Backend Deployment (GitHub actions side)

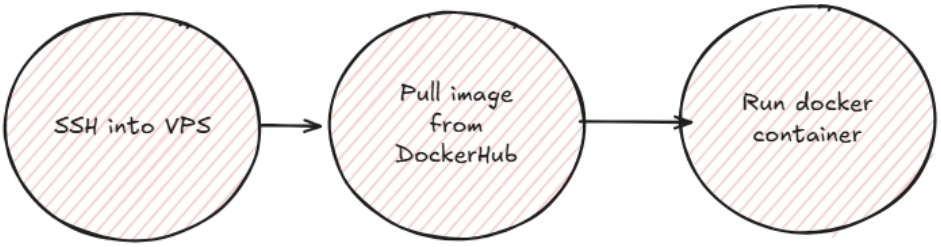

Figure 3.6: Backend Deployment (VPS side)

Chapter 4

Challenges

Throughout developing Planify platform, we have faced multiple challenges to meet the set requirements to best deliver on the proposed solution, they are as follows:

4.1 SSL Certificate

The need for authenticated requests for planify required managing and issuing an SSL certificate from a vendor.

The setup required some complexity to create a private/public key pairs, along with creating a Certificate Signing Request (CSR) containing all required details.

And then sending the certificate files to VPS using SFTP to ssh into the vps.

4.2 Multi-tenancy

The decision to implement multi-tenancy by passing the tenant name as a sub-domain in place of "www" required for a more sophisticated SSL certificate.

A normal one didn't support multiple sub-domains and wild card support, which we solved by finding a different SSL and issue another Certificate Signing Request (CSR) to be able to use it.

4.3 Multiple backend systems

The use of a separate isolated recommendation engine added complexity for deployment on the server. Planify backend needed to use the engine to get recommendation while both are hosted on the same Virtual Private Server (VPS).

We solved this by using NGINX http server as a reverse proxy, with a configuration file to expose specific routes that map requests to designated backend applications to handle, all while making sure requests are encrypted by the use of SSL certificate.

Chapter 5

Results and Discussion

The implemented Planify platform successfully delivers what it promised in the introduction. This section presents the individual components. the platform needs to meet its requirements. The code for all componenets can be found in GitHub in the references section.

- **Planify-BE**: Backend application responsible for business logic of Planify
- **Planify-FE**: Frontend application facing users of the platform
- **Planify-Mobile**: Mobile application for actions that are most suitable for a mobile device
- **Recommendation Engine**: a Python service with a responsibility of recommending roadmaps based on a given input

In this section, we discuss the finished product based on the methodology explained. We will cover how the finished product looks and how it is used to achieve the desired output of this project. For ease of reading, screens will be distributed among flows users can do, and in each flow, we cover multiple pages so that we show each page and its usage in the finished application. We will start with the web application and then in the next step, cover what the mobile application has to offer.

5.1 Web Application User Interface

5.1.1 Sign-up

Users enter planify by signing up (Figure 5.1), a user puts in the credentials, and he will receive an email as shown in (Figure 5.2) to verify his account, when clicking on verify, we will be able to enter into the home page (Figure 5.3).

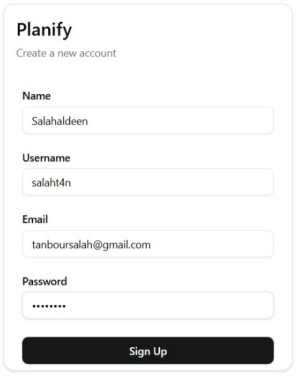

Figure 5.1: Sign up page

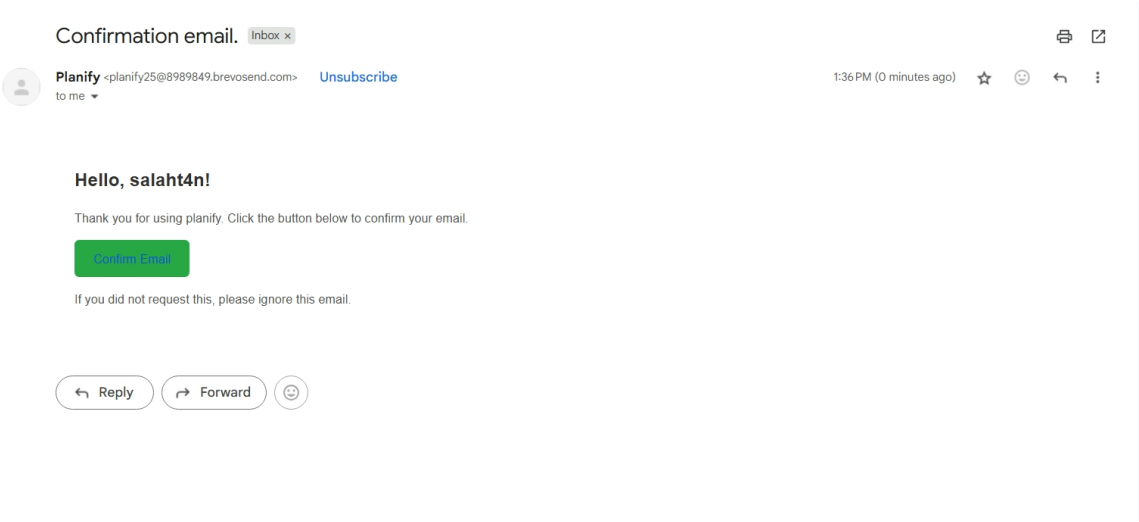

Figure 5.2: Email to verify

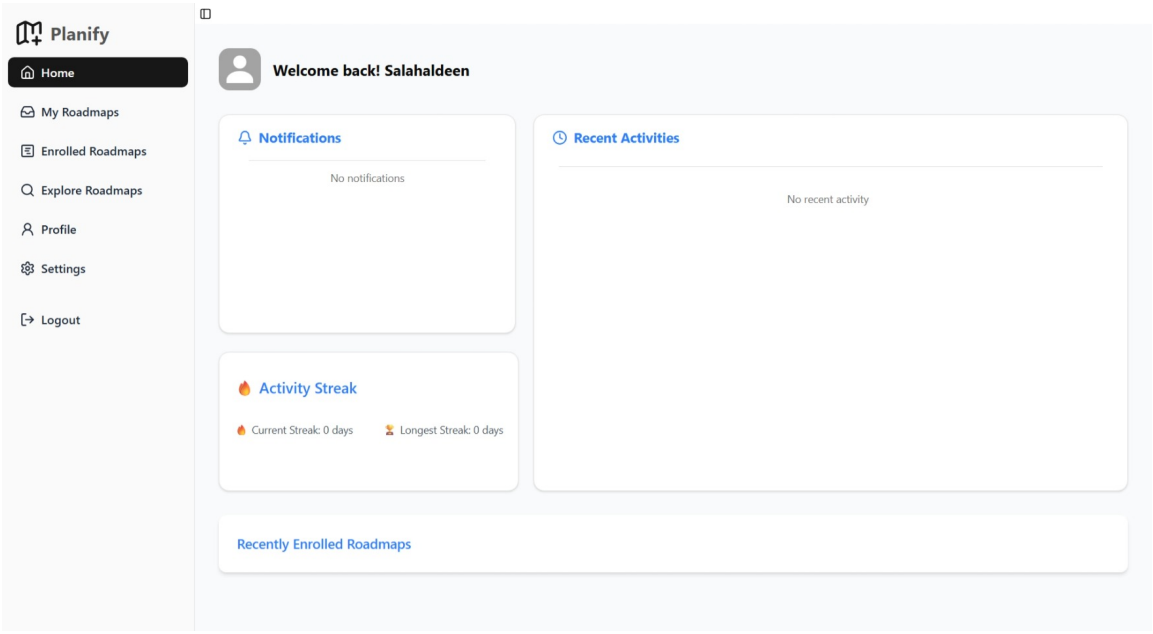

Figure 5.3: Planify Homepage

We can find Activity Streak (Figure 5.4)to show the streak of days the user has been doing progress, and a recently enrolled roadmaps section to quickly jump and start learning (Figure 5.5).

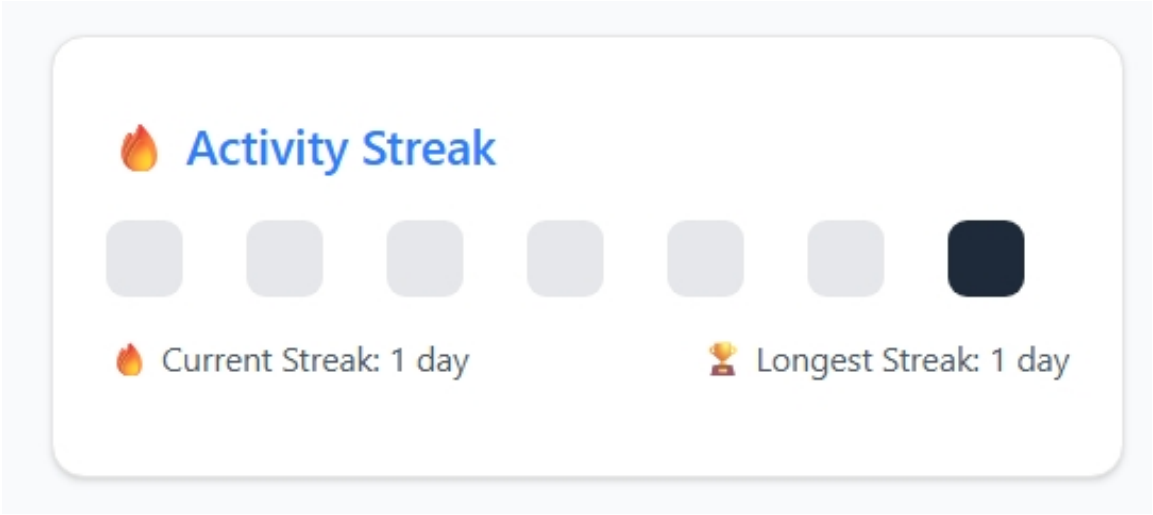

Figure 5.4: Activity Streak

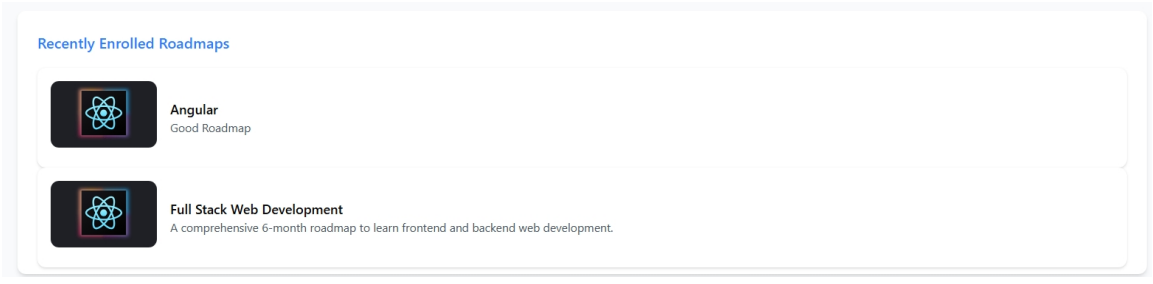

Figure 5.5: Recently Enrolled Roadmaps

5.1.2 Reset Password

The password reset flow in our application is designed to be intuitive and user-friendly, guiding the user through each step with clear instructions. As shown in Figure 5.6, the user initiates the process by clicking on "Forgot Password," which opens a modal prompting them to enter their registered email address. Once submitted, the system displays a confirmation message (Figure 5.7), informing the user that a reset link has been sent to their email. Upon receiving the email and clicking the provided link, the user is redirected to the verification page depicted in Figure 5.8, where the system validates the token and prepares the next step. Finally, as shown in Figure 5.9, the user is presented with a form to securely enter and confirm their new password, completing the reset process.

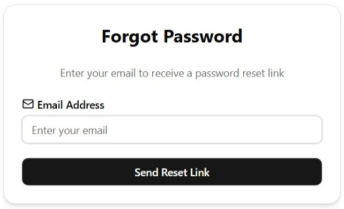

Figure 5.6: Password reset modal prompting for email

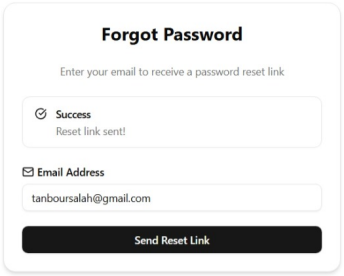

Figure 5.7: Confirmation message after requesting password reset

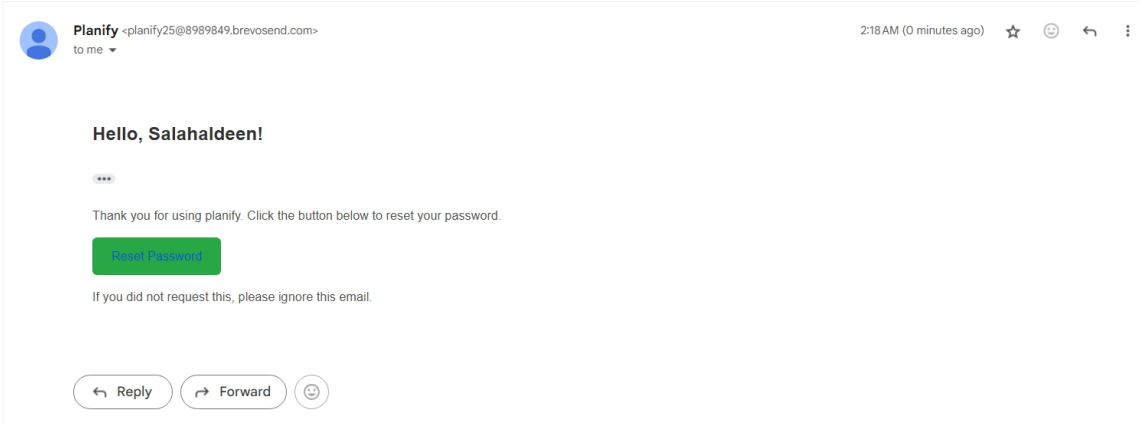

Figure 5.8: Email verification page after clicking reset link

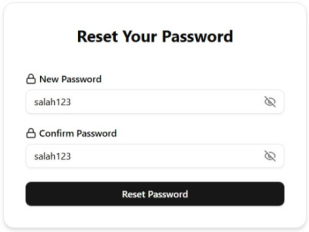

Figure 5.9: Form to enter and confirm new password

5.1.3 Create Roadmap

The roadmap creation flow allows users to easily design and manage structured learning paths. As shown in Figure 5.10, the process begins when the user opens a modal to input basic roadmap information such as title and description. Upon submission, the roadmap becomes visible in the "My Roadmaps" page (Figure 5.11), where users can view all their roadmaps. Figure 5.12 shows the interface after a roadmap is created. Using the actions menu (Figure 5.13), users can edit or delete a roadmap. By clicking on a roadmap, users enter the builder view (Figure 5.14), where they can start adding content.

In the builder, users can add nodes (Figure 5.15), connect them with edges (Figure 5.16), and modify node properties like color and label (Figure 5.17). Opening a node reveals a markdown editor modal (Figure 5.19), where users can write content or use slash commands to insert elements. Figure 5.20 shows the live preview mode of this content.

Once published, roadmaps appear on the explore page (Figure 5.27) for other users to view. Additionally, a new activity entry is generated on the home page 5.18, indicating that the user has created a new roadmap.

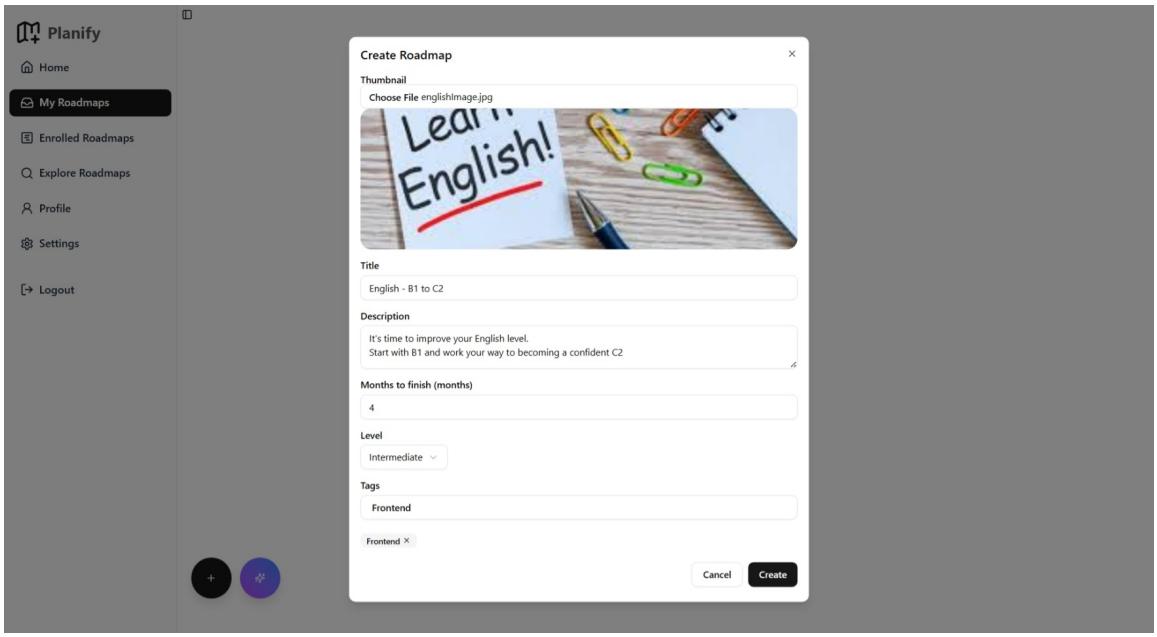

Figure 5.10: Create roadmap modal

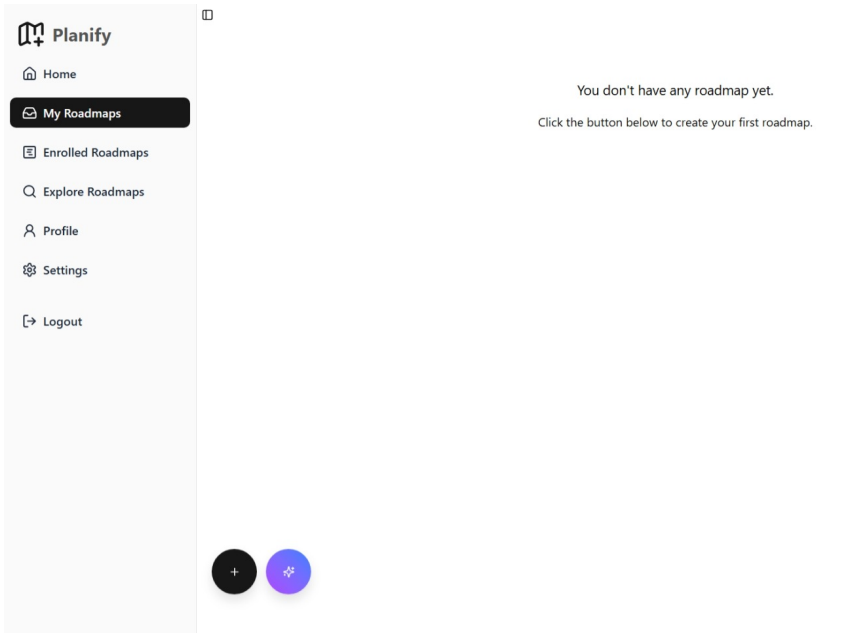

Figure 5.11: My Roadmaps page showing user's created roadmaps

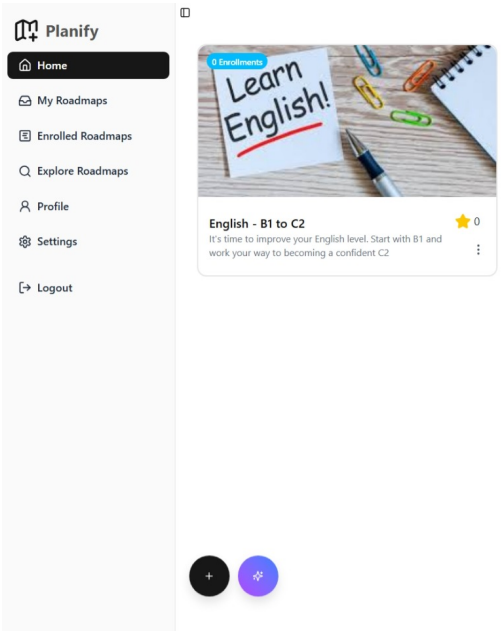

Figure 5.12: My Roadmaps page after creating a new roadmap

Figure 5.13: Actions available for each roadmap

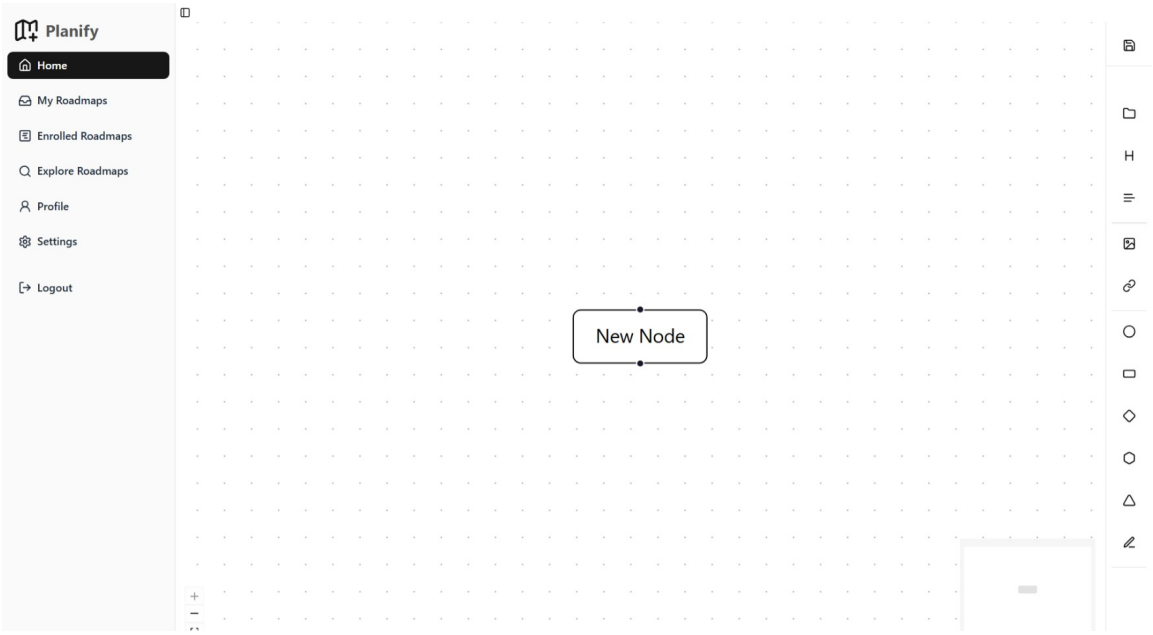

Figure 5.14: Empty roadmap builder view

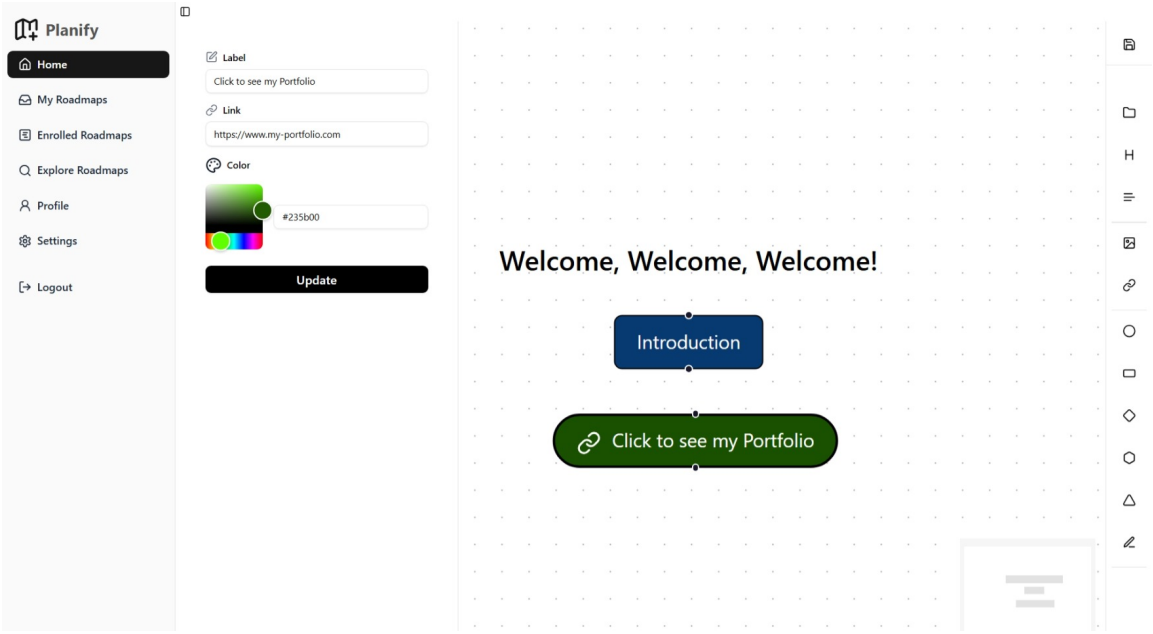

Figure 5.15: Adding more node types to the builder

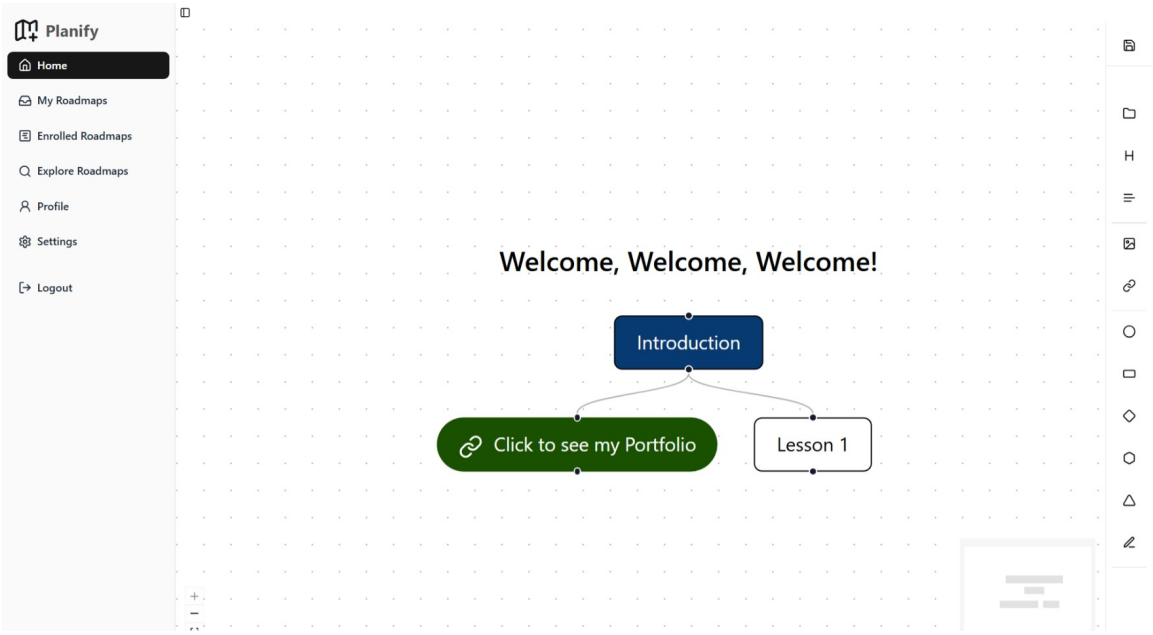

Figure 5.16: Connecting nodes with edges

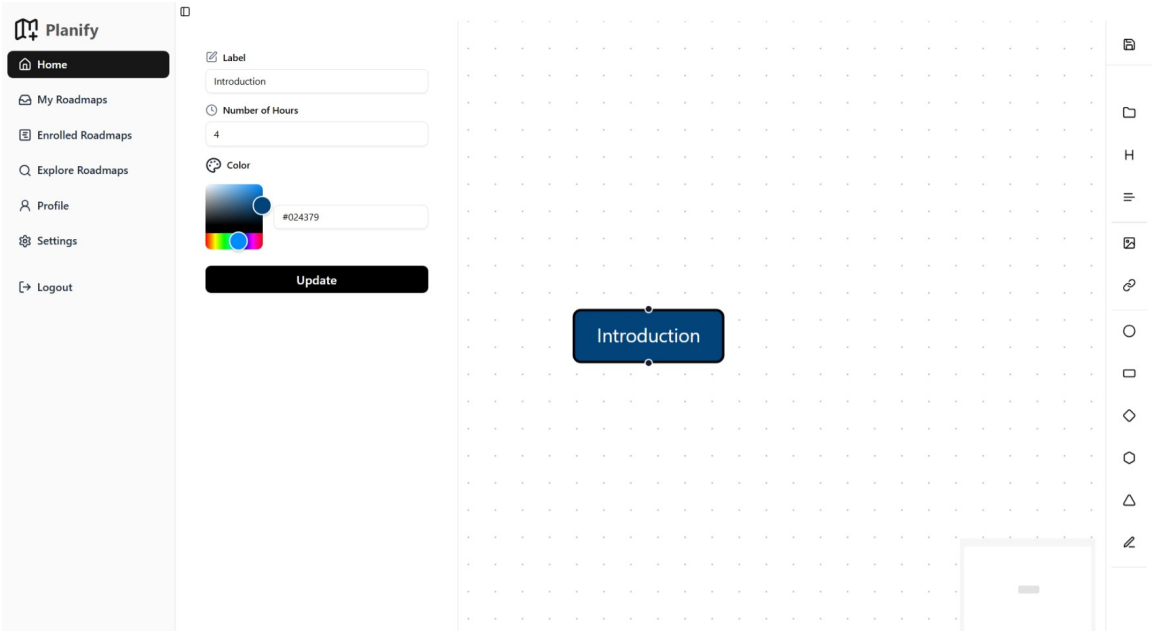

Figure 5.17: Changing node properties like color or label

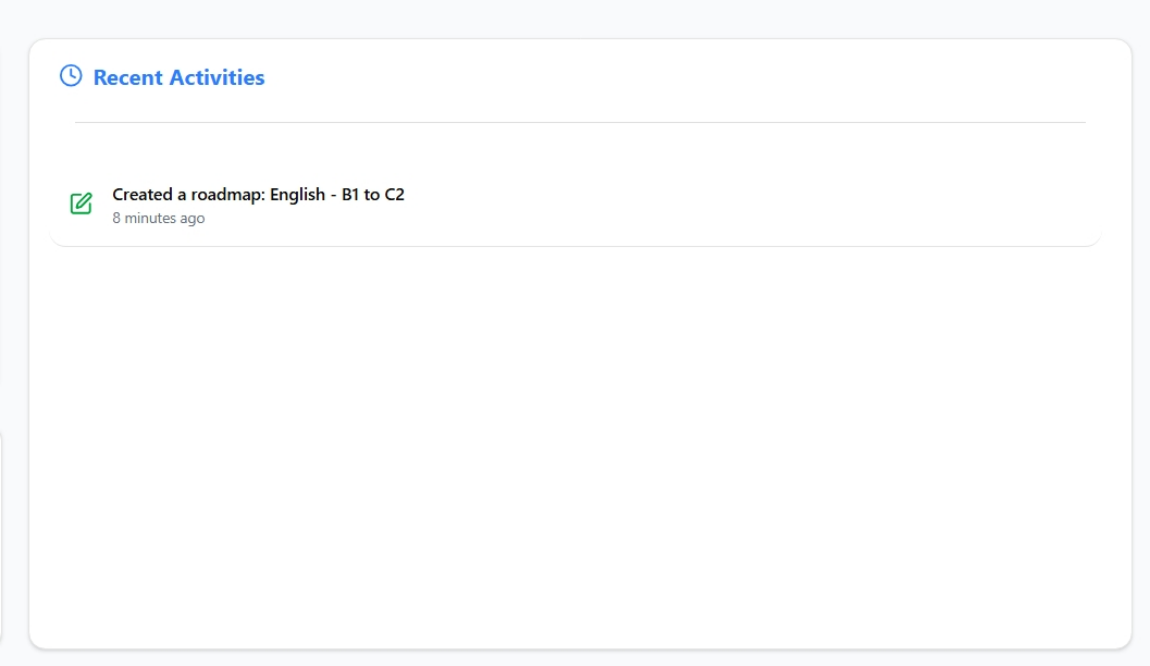

Figure 5.18: User Activities in Home page

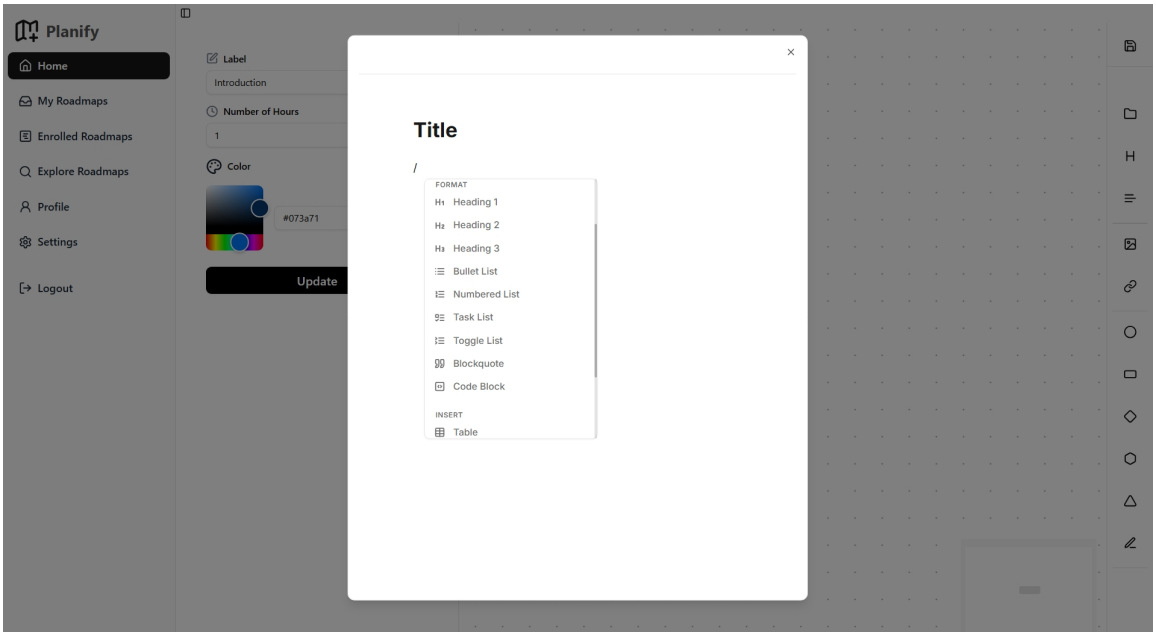

Figure 5.19: Markdown text editor for node content

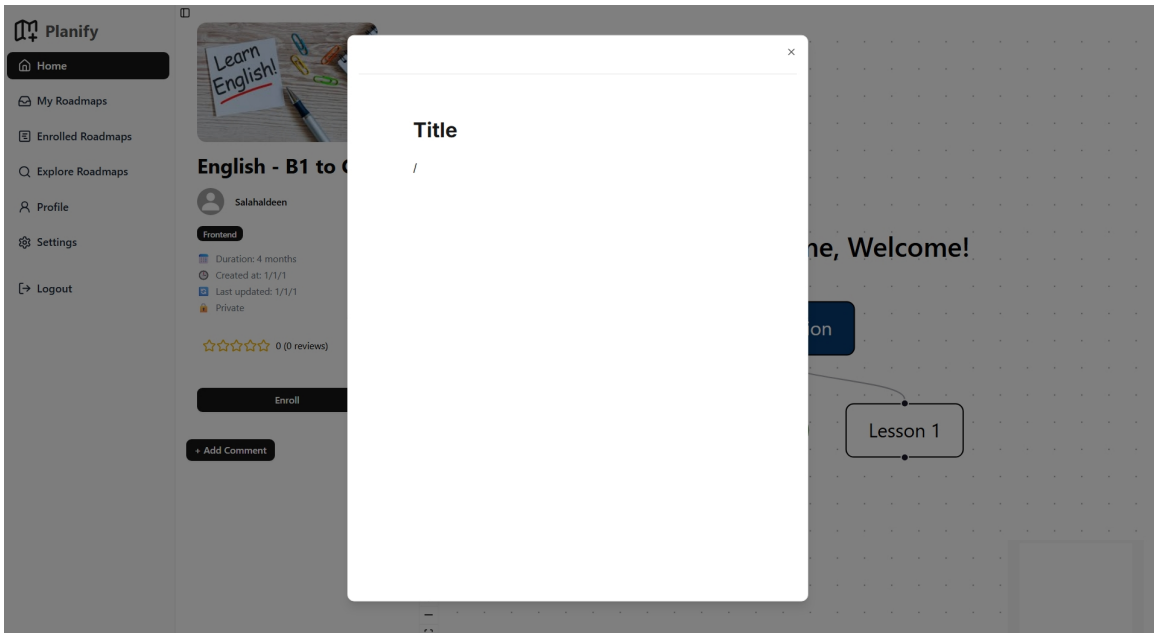

Figure 5.20: Live preview of the markdown editor

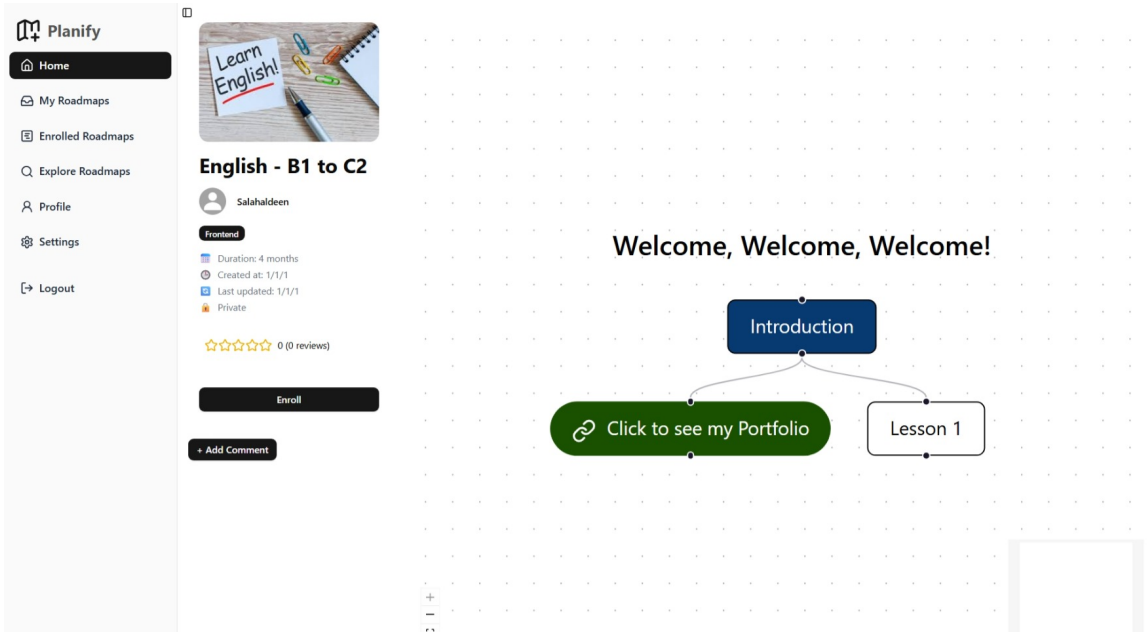

Figure 5.21: Roadmap preview as seen by other users

5.1.4 Search & Explore Roadmaps

Users can discover public roadmaps through the Explore page, shown in Figure 5.22, where they can browse, search, or filter content based on their interests. When a roadmap catches their attention, clicking it navigates them to the Roadmap Details page (Figure 5.23), where they can view the full content and structure of the roadmap.

To engage with the roadmap, users can leave comments (Figure 5.24), fostering interaction and feedback. Additionally, clicking on the author's name allows users to visit the author's profile (Figure 5.25), which displays other roadmaps created by that user—encouraging exploration of related content.

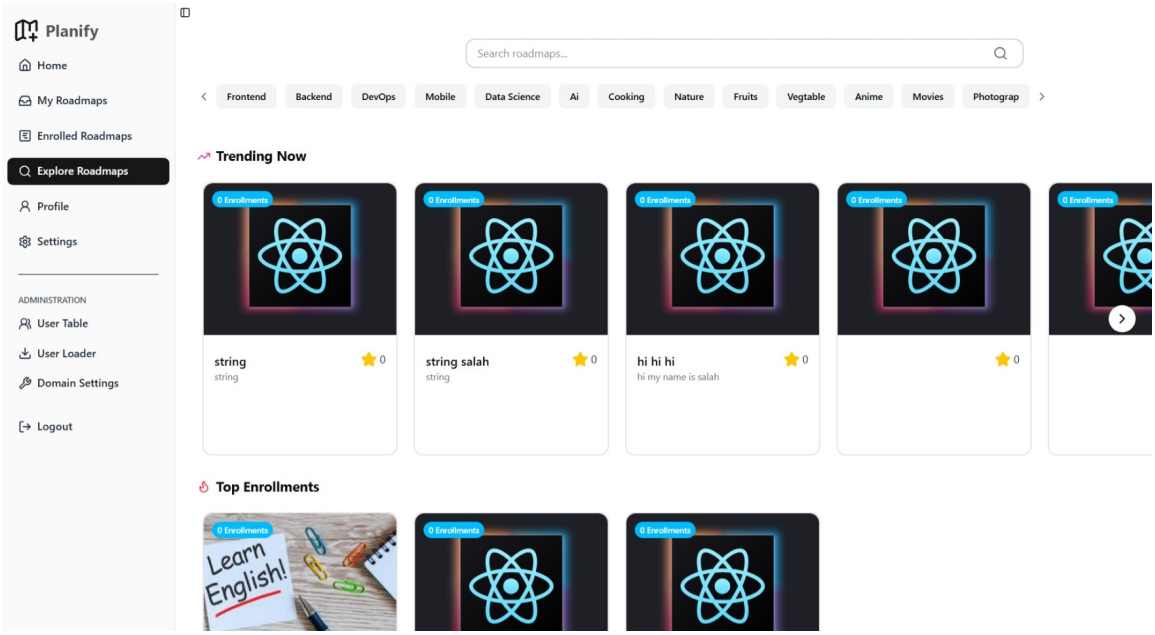

Figure 5.22: Explore page showing public roadmaps with search and filter options

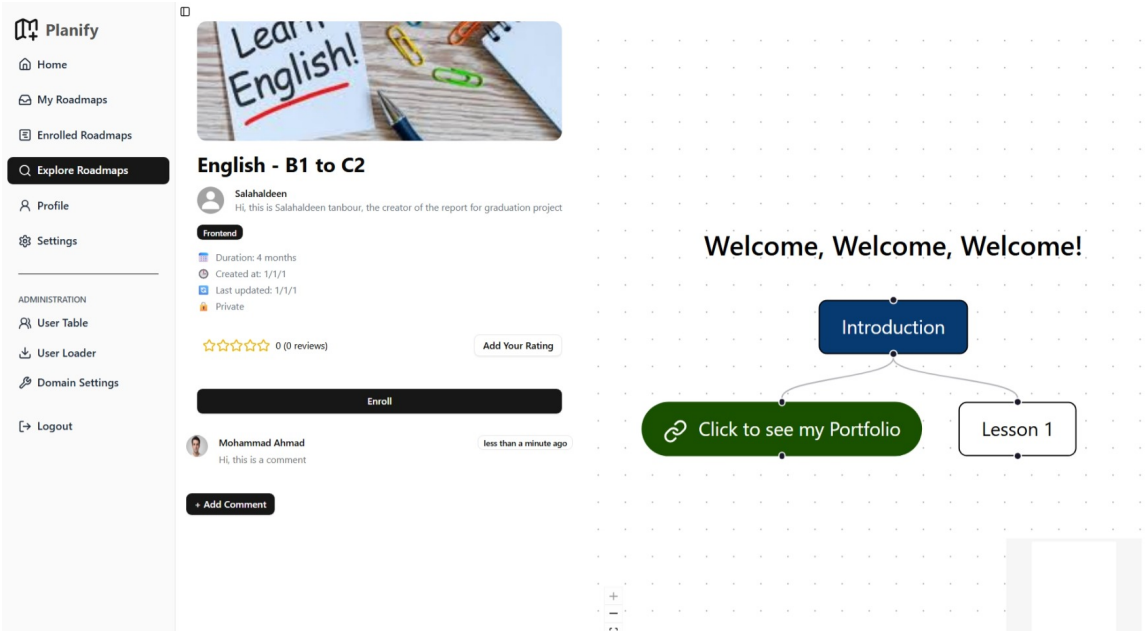

Figure 5.23: Detailed view of a selected roadmap

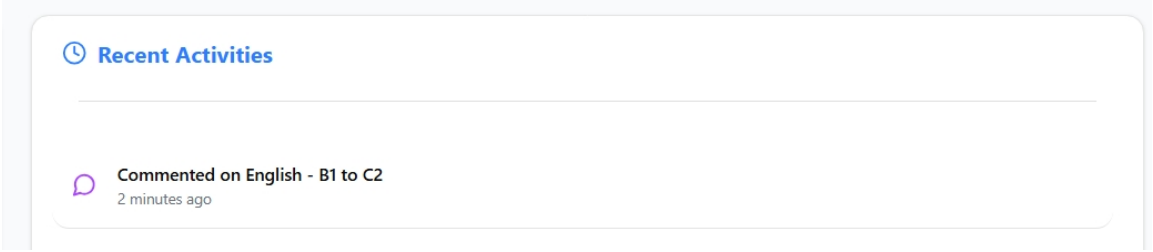

Figure 5.24: User adding a comment on a roadmap

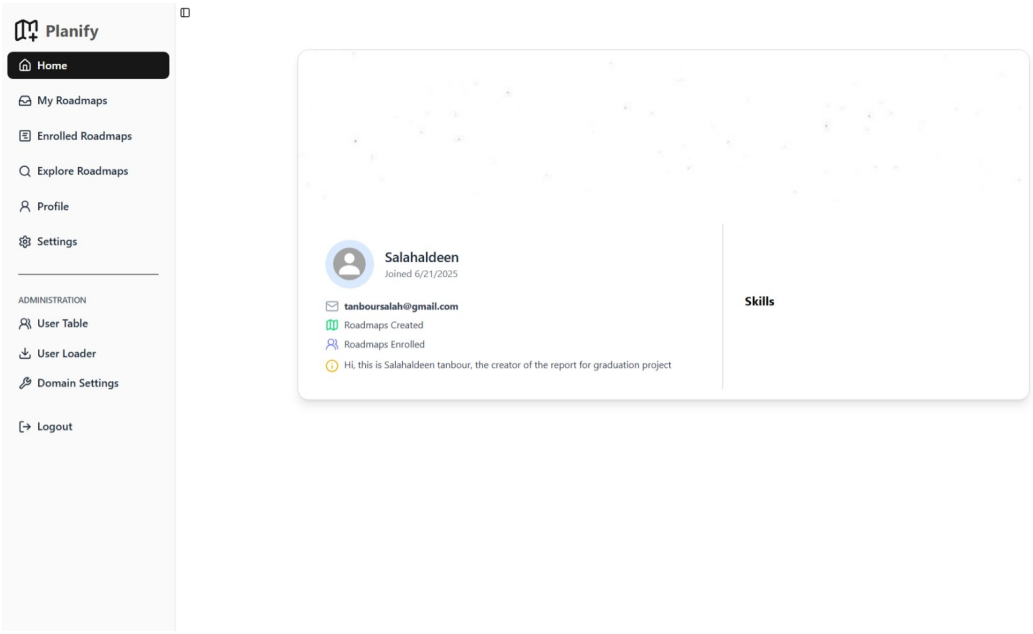

Figure 5.25: Author profile displaying other published roadmaps

5.1.5 Display Enrolled Users Progresses

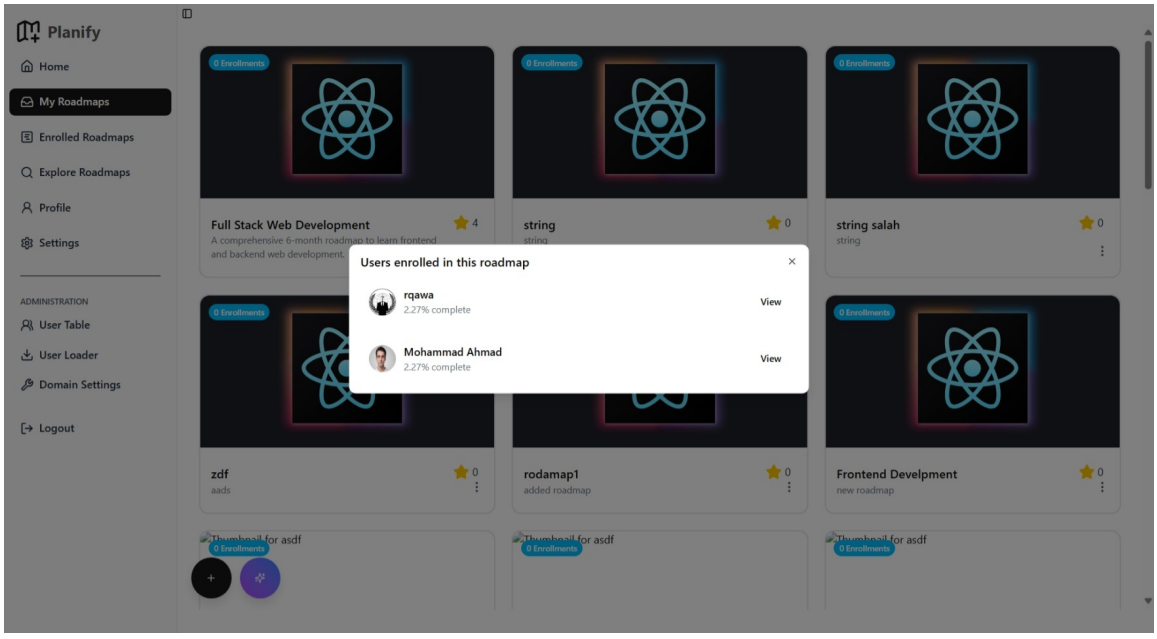

Figure 5.26: Roadmap preview as seen by other users

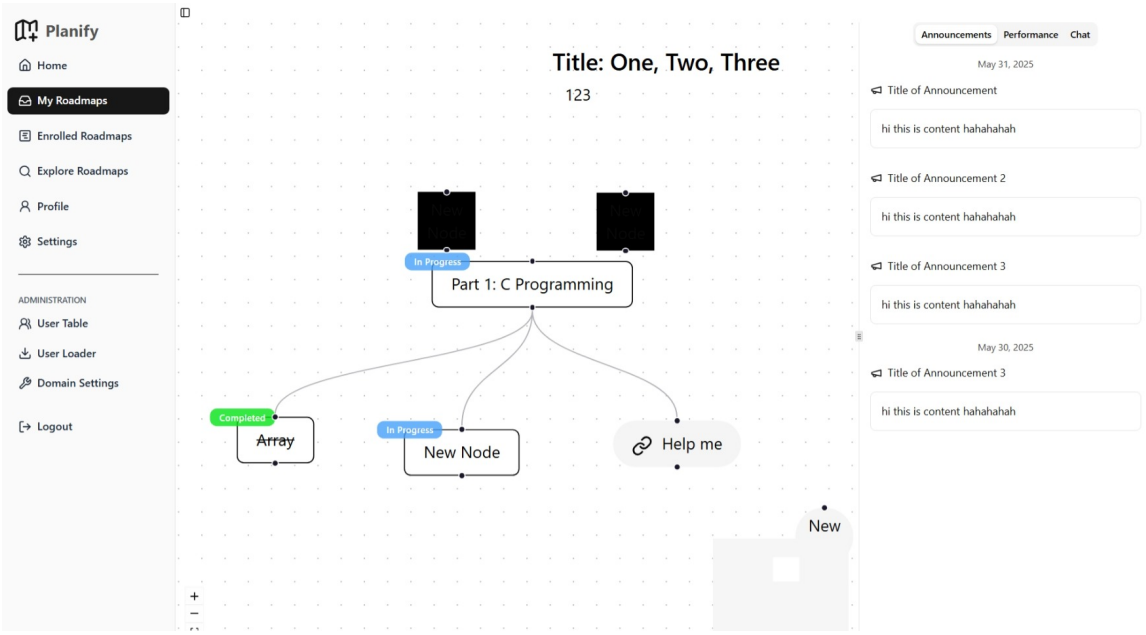

Figure 5.27: Roadmap preview as seen by other users

5.1.6 Enroll in Roadmap

In the "My Roadmaps" section, creators can view a list of all users enrolled in a specific roadmap. As shown in Figure 5.28, this table displays each user's basic information along with options to inspect their progress. By selecting a user, the creator is taken to a view inside the builder (Figure 5.29), where they can track that user's progress through the roadmap nodes—offering insight into engagement and completion rates.

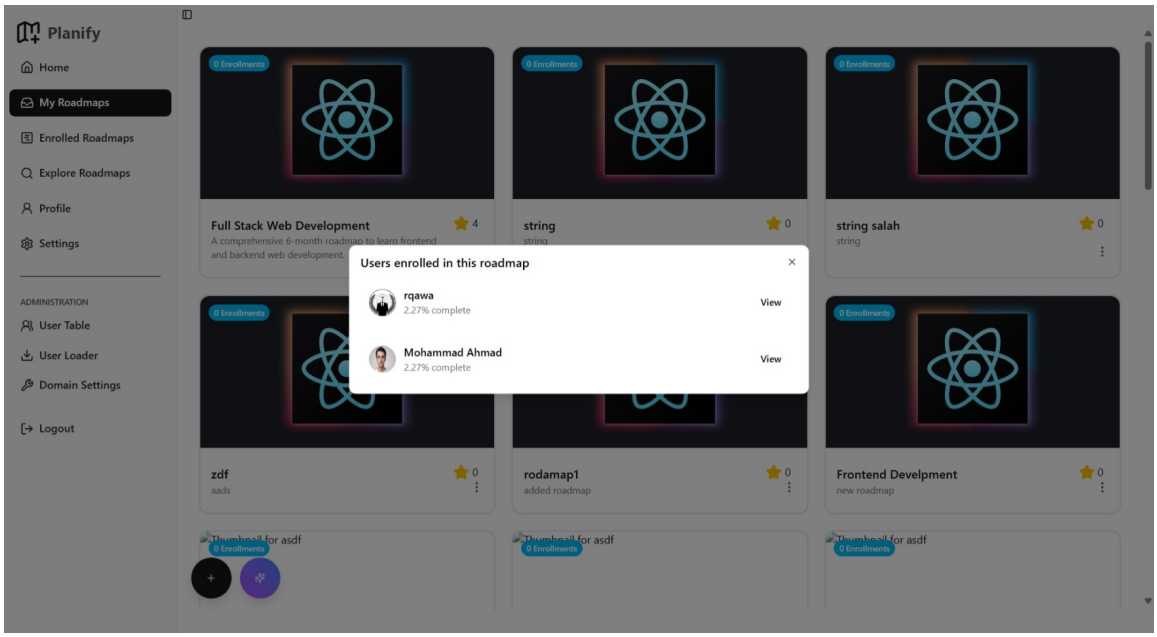

Figure 5.28: Table of users enrolled in a roadmap

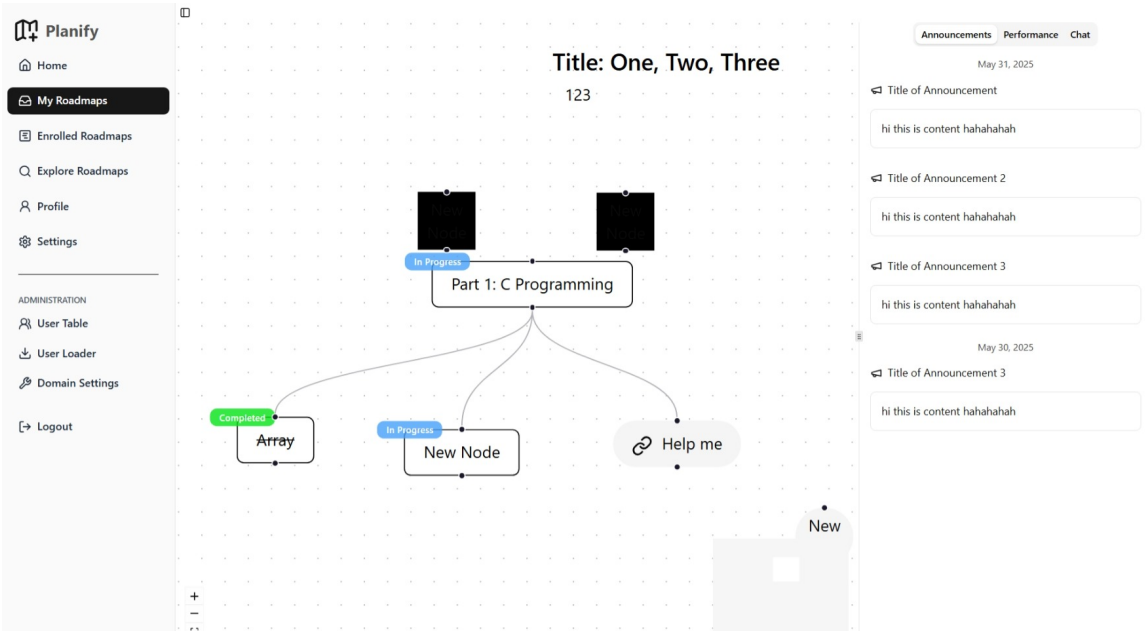

Figure 5.29: Inspecting an enrolled user's progress in the builder

5.1.7 Enrollment & Tracking Progress

The enrollment page enhances user engagement through collaborative and competitive features. On the sidebar, users can view announcements from the roadmap creator (Figure 5.30), keeping them informed about updates or guidance. To foster a sense of competition and motivation, a leaderboard (Figure 5.31) ranks users based on their progress and activity. Additionally, a real-time group chat (Figure 5.32) allows all enrolled users to communicate, share tips, and ask questions—turning the learning experience into a more interactive and social environment.

Announcements | Performance | Chat

May 31, 2025

Title of Announcement

hi this is content hahahahah

Title of Announcement 2

hi this is content hahahahah

Title of Announcement 3

hi this is content hahahahah

May 30, 2025

Title of Announcement 3

hi this is content hahahahah

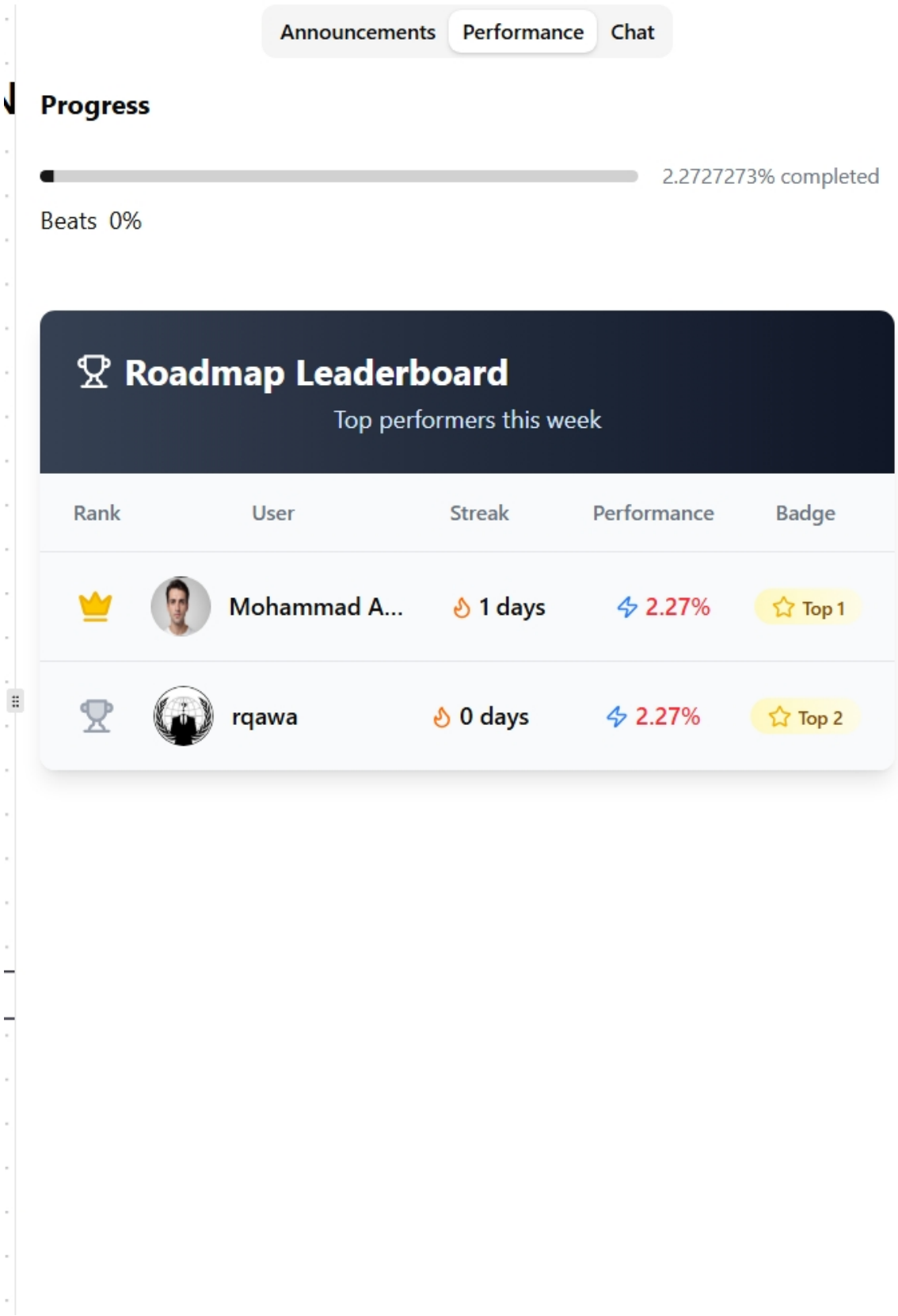

Figure 5.31: Leaderboard comparing enrolled users' progress

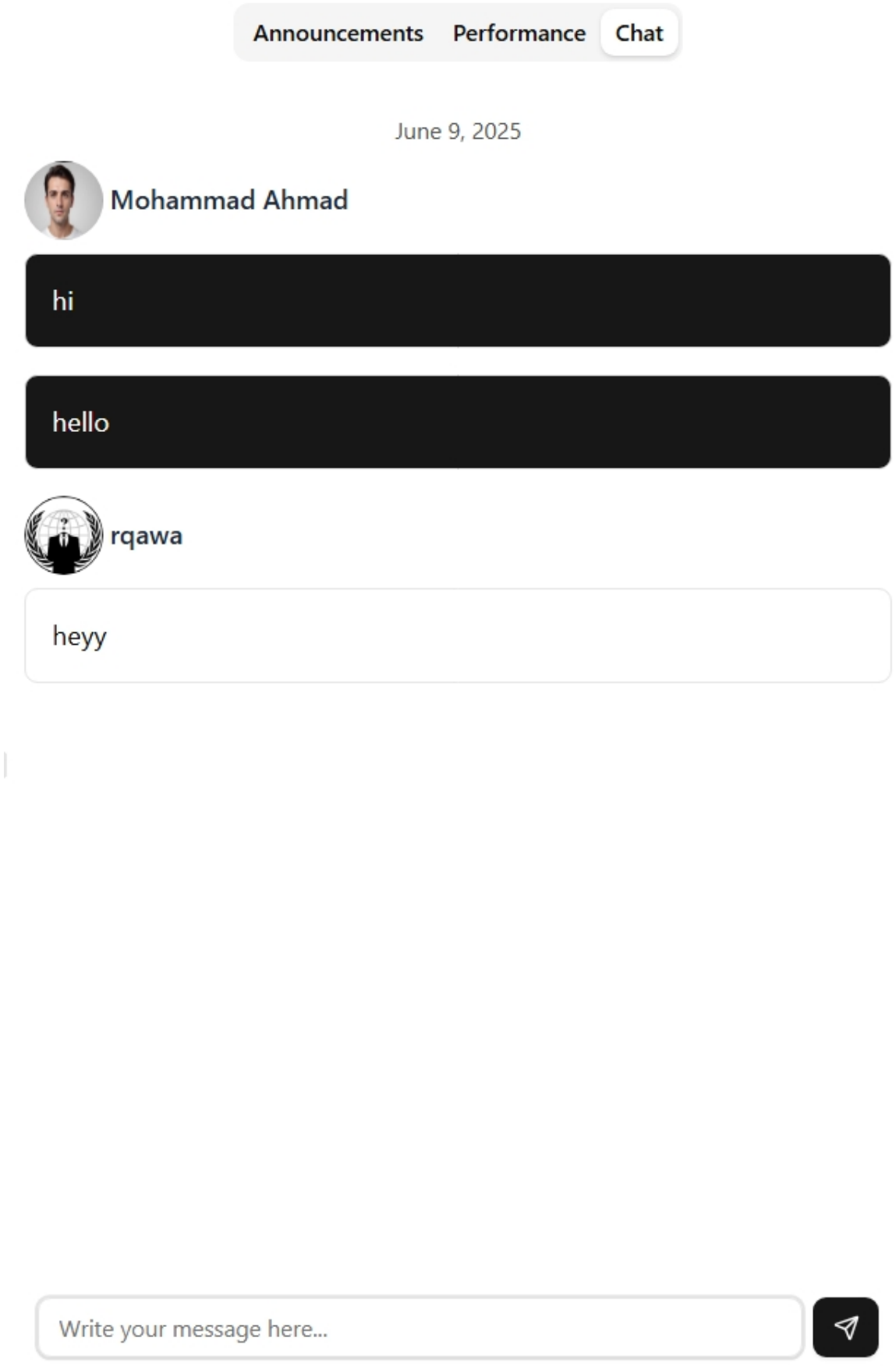

Figure 5.32: Real-time group chat for enrolled users

5.1.8 Change Profile Info

Users can manage their personal information through the profile page, shown in Figure 5.33, where they can view their public details and added skills. By clicking the edit button, they are taken to a form (Figure 5.34) where they can update their profile and add relevant skills. These skills are later used in the recommendation system on the Explore page to suggest roadmaps that align with the user's interests and background.

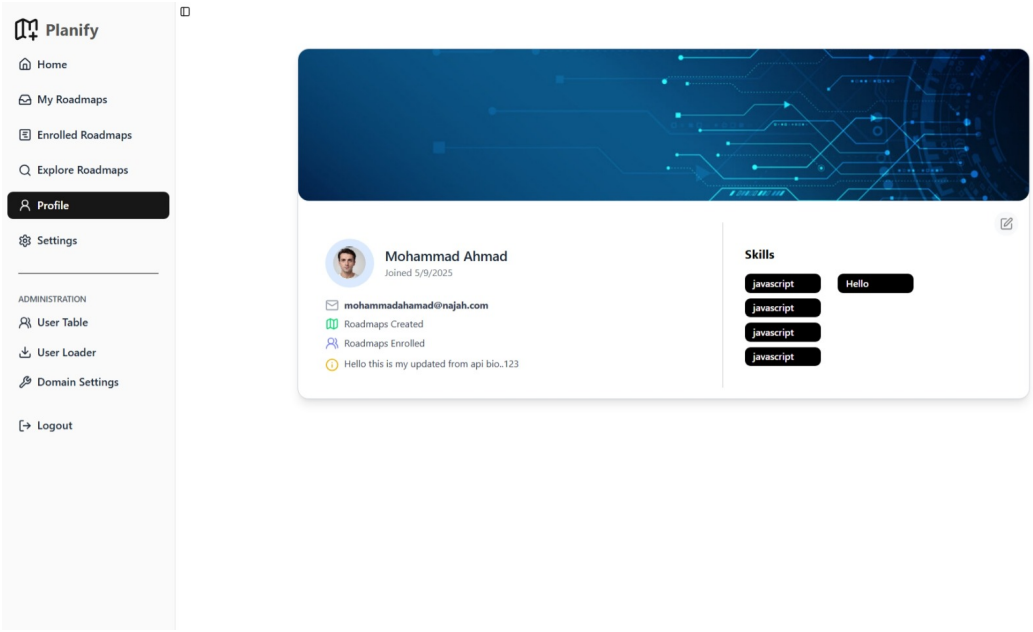

Figure 5.33: User profile page displaying public information and skills

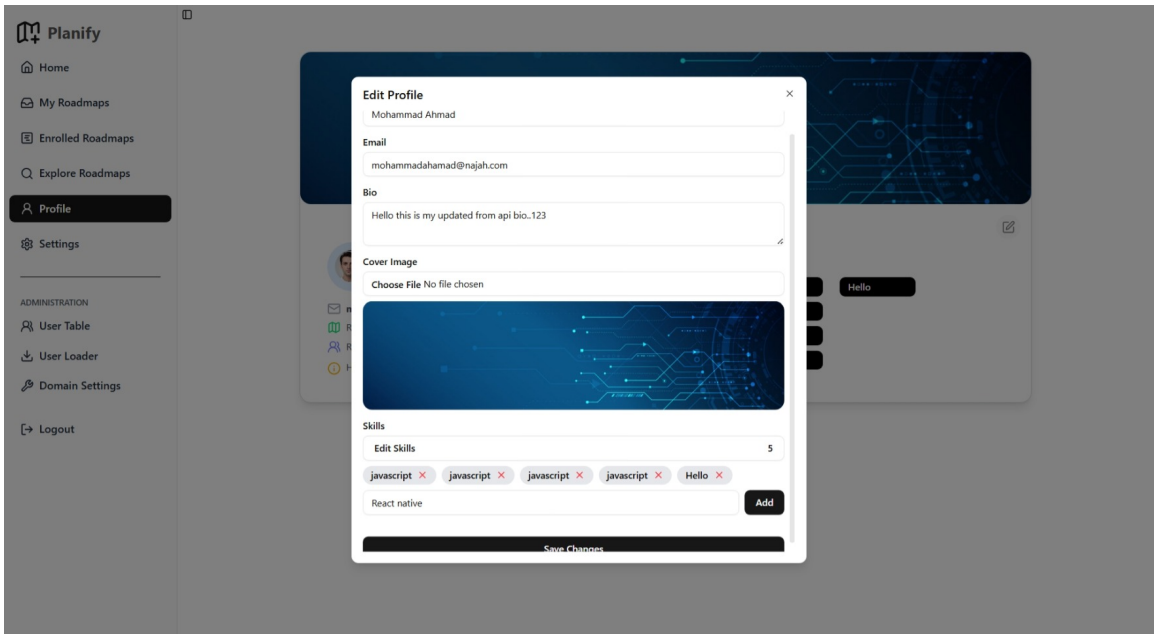

Figure 5.34: Edit profile form for updating details and adding skills

5.1.9 Customize Application with Domain Settings

The platform allows administrators to customize domain-specific settings through the client settings page (Figure 5.35). This interface includes checkboxes and input fields for toggling features such as enabling comments, chat access, and system-wide tags, as well as updating the display title. After making changes, the effects are immediately visible across the application—for example, as shown in Figure 5.36, the updated title appears on the sign-in page, reflecting the applied customization.

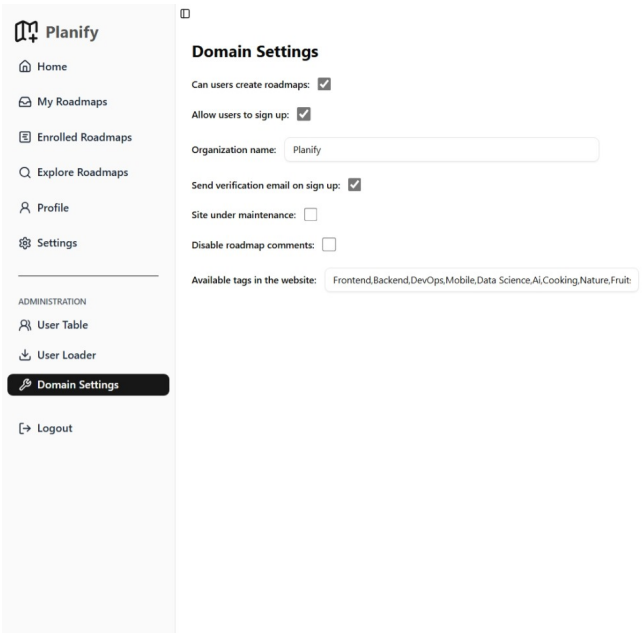

Figure 5.35: Client settings page for configuring domain-specific options

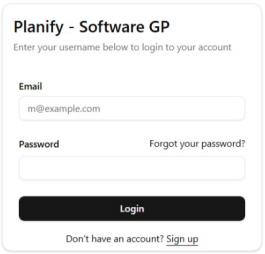

Figure 5.36: Sign-in page reflecting updated domain title

5.1.10 Users Loader

For organizational administrators, the platform provides a user import feature to streamline onboarding. As shown in Figure 5.37, the interface includes options to download a CSV template and upload a completed one. Figure 5.38 shows the download action, which provides a blank template (Figure 5.39). After filling in user information (e.g., name, email), the completed template resembles Figure 5.40.

Once the template is uploaded (Figure 5.41), the system processes the entries and adds users to the organization. The updated user table, now including the imported user, is shown in Figure 5.42. Finally, as part of the onboarding process, the new user receives an email with a password reset link (Figure 5.43), allowing them to access the system securely.

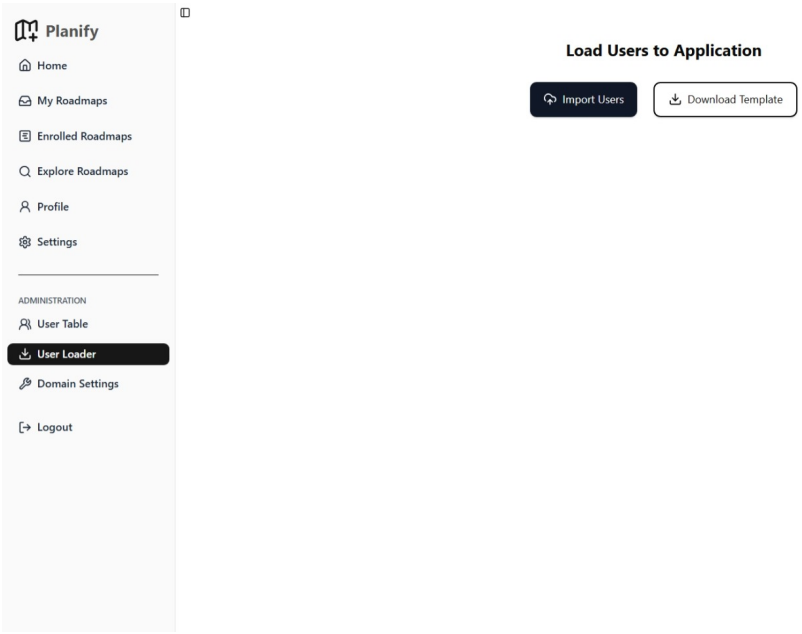

Figure 5.37: User import page with buttons to download template and upload filled file

assets/screenshots/user-loader/2.jpeg

Figure 5.38: Downloading the user import template

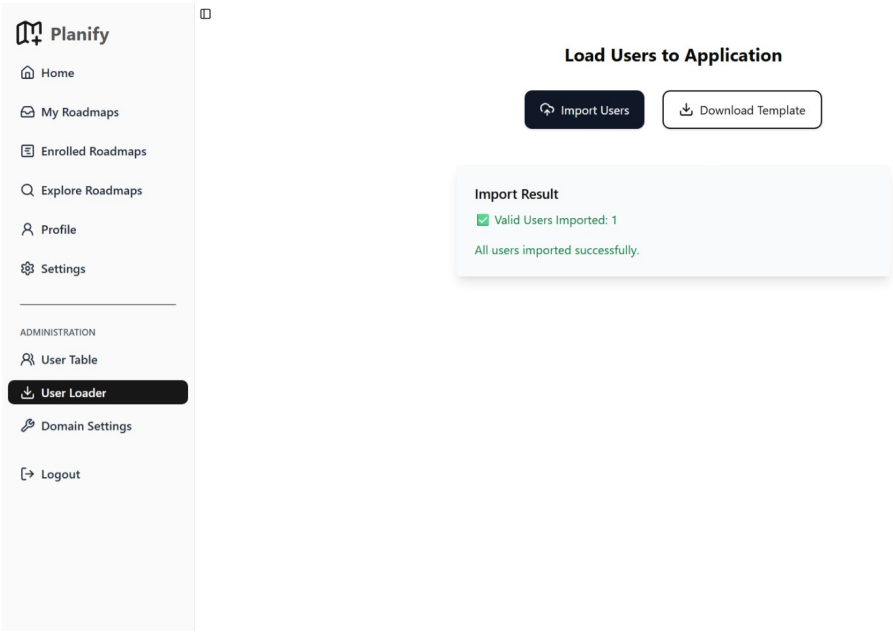

Figure 5.39: Empty CSV template for user import

assets/screenshots/user-loader/4.jpeg

Figure 5.40: CSV template with user information filled out

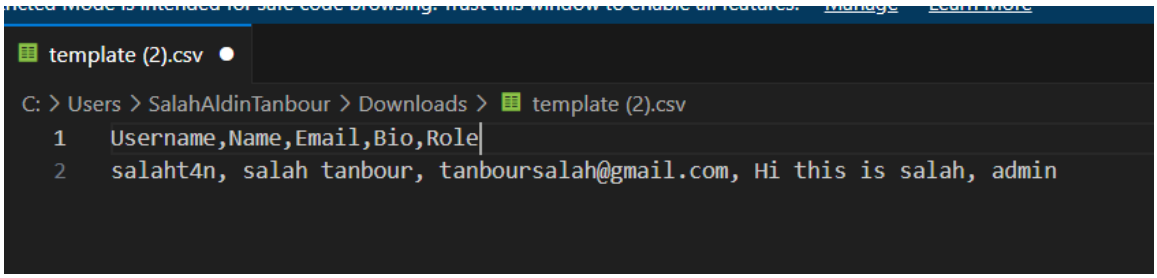

Figure 5.41: Uploading the completed CSV template

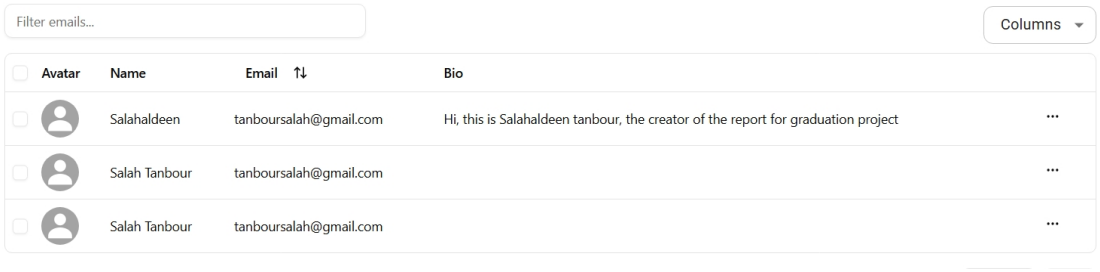

Figure 5.42: User successfully added to the user table

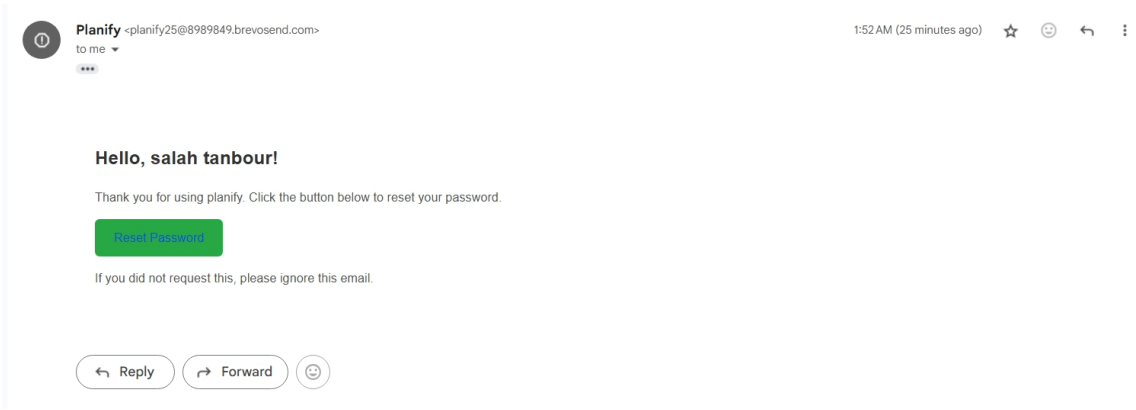

Figure 5.43: New user receives email with reset password link

5.1.11 Users Table

The users table page provides administrators with a comprehensive, paginated overview of all users in the system (Figure 5.44). The table supports customization, allowing admins to select or deselect columns to tailor the view to their needs (Figure 5.45). Figure 5.46 shows a snapshot of the populated user list, enabling efficient user management and quick access to relevant user data.

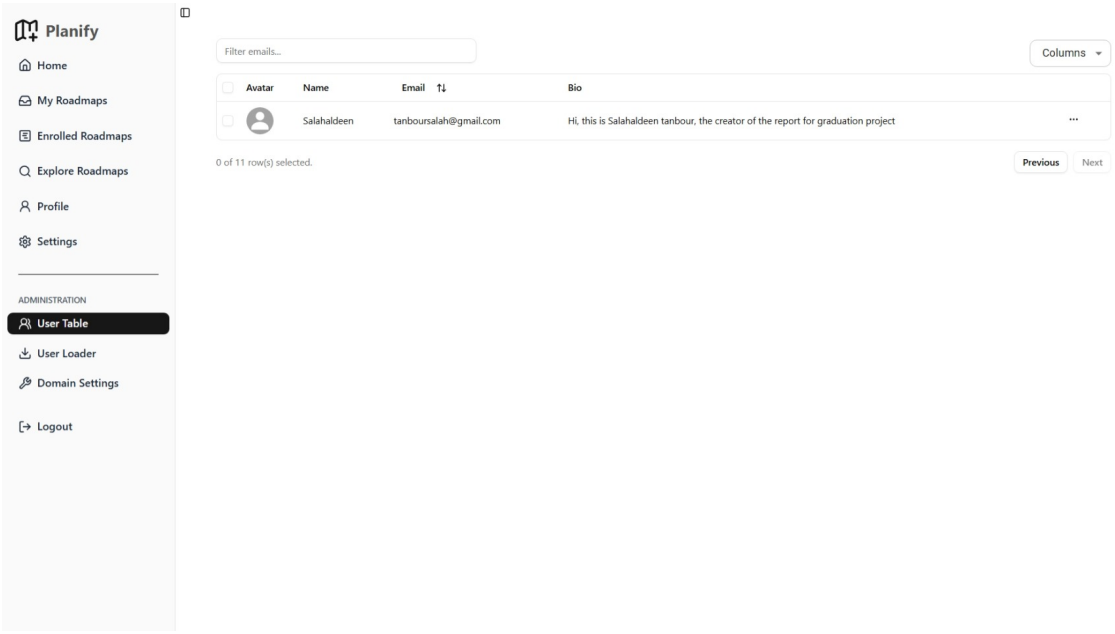

Figure 5.44: Users table page overview

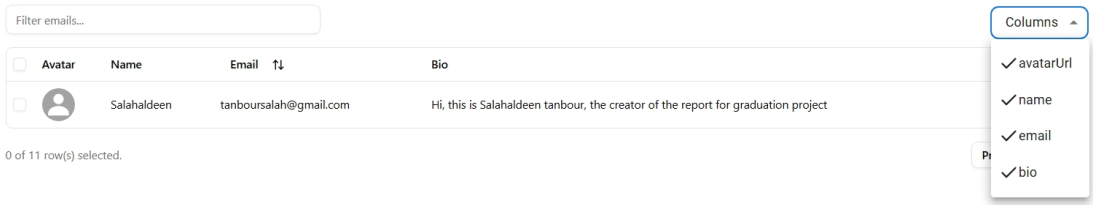

Figure 5.45: Customizing visible columns in the users table

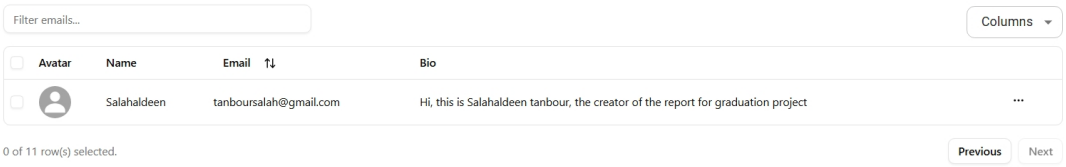

Figure 5.46: Paginated list of users in the system

5.1.12 Generate Roadmap with AI

Chapter 6

Conclusion and Future Work

In today's age, and with the quantity of free and available learning resources on the internet, people can get overwhelmed and struggle to find a place to start. And this increases the benefits structured learning can have. Roadmaps are an essential part of structured learning, and can drive learning speeds to a significant degree.

In this research, a proposed solution was the creation of a web application, Planify, to serve as the platform that hosts many roadmaps across different domains, with the ability for users to create their own roadmaps and share them with other learners on the platform. In addition to supporting individual needs, it also delivers on organizational level of needs by utilizing sub-domains and isolated multi-tenancy concept to a high degree, to deliver an isolated experience for the company that benefits better user experience and security.

While Planify delivers a comprehensive mvp, there is still a backlog of features that could greatly impact and improve on the current implementation, these include:

- Expanded customization in the builder
- Filtering of comments content for bad words
- More robust reviewing system

Appendix A

Appendix: NGINX Configuration

This appendix includes the configuration used to configure NGINX as a reverse proxy.

```
server {
    listen       80;
    listen  [::]:80;
    server_name  localhost;


    location /recommend {
        proxy_pass http://62.171.153.7:8000/recommend;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    location / {
        return 301 https://$host$request_uri;
    }
}

server {
    listen       443 ssl;
    listen  [::]:443 ssl;
    server_name  localhost;

    ssl_certificate      /usr/share/nginx/ssl/chain.pem;
    ssl_certificate_key  /usr/share/nginx/ssl/server.key;

    #access_log  /var/log/nginx/host.access.log  main;

    location /google {
        return 302 https://www.facebook.com;
    }

    location /recommend {
```

```
        proxy_pass http://62.171.153.7:8000/recommend;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    location /swagger {
        proxy_pass http://62.171.153.7:8080/swagger;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    location /api {
        proxy_pass http://62.171.153.7:8080/api;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    location /static {
        proxy_pass http://62.171.153.7:8080/static;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    location / {
        root /usr/share/nginx/html;
        index index.html;
        try_files $uri /index.html;
    }
}
```

Appendix B

Appendix: Project Management

This appendix details the project management approach and methodologies used in developing the Plahify platform.

B.1 Development Methodology

The project followed an Agile development methodology with two-week sprint cycles. Key aspects include:

- Regular sprint planning and retrospective meetings
- Daily standup meetings for progress tracking
- Continuous integration and deployment
- Iterative feature development and testing

B.2 Tools and Practices

The following tools and practices were used to effectively manage the project:

- **Version Control**: Git with GitHub for source code management
- **Communication**: Discord for team communication
- **CI/CD**: GitHub Actions for Automated Builds and Deployments
- **Virtual Private Servers**: For front-end and back-end deployment.

B.3 Risk Management

Key risks were identified and managed throughout the project:

- **Technical Risks**:
  - Isolated Multi-tenancy and SSL Certificate support
  - Performance scalability

  - Cross-platform compatibility

- **Mitigation Strategies**:
  - Early prototyping of critical features
  - Regular performance testing
  - Cross-platform testing automation

Appendix C

Appendix: Development Workflow

The complete source code for the Planify platform is organized across multiple repositories on GitHub (***).

The development workflow includes:

- Feature branches for development
- Pull request reviews for code quality
- Automated deployment to VPS using CI/CD pipelines

For detailed setup and contribution guidelines, refer to each repository's README file.

References

[1] Wikipedia contributors, "Instructional design — wikipedia, the free encyclopedia," 2023, [Accessed: 2025-06-27]. [Online]. Available: https://en.wikipedia.org/wiki/Instructional_design

[2] ——, "Instructional scaffolding — wikipedia, the free encyclopedia," 2023, [Accessed: 2025-06-27]. [Online]. Available: https://en.wikipedia.org/wiki/Instructional_scaffolding

[3] M. Valle Torre *et al.*, "The sequence matters in learning: A systematic literature review," *arXiv preprint*, 2023. [Online]. Available: https://arxiv.org/abs/2308.01218

[4] C. De Smet, B. De Wever, and T. Schellens, "The design and implementation of learning paths in a learning management system," *Interactive Learning Environments*, vol. 24, no. 6, pp. 1076–1096, 2014.

[5] N. Ovtšarenko, "Generation of a learning path in e-learning environments: Literature review," *International Journal of Education and Learning Systems*, vol. 8, pp. 1–8, 2023. [Online]. Available: https://www.researchgate.net/publication/369959738