# Capstone Step 2
## Project Proposal

We have broken down the Capstone Project into easy-to-follow steps. Let's dive into Step two of your Capstone Project. Step Two is all about fleshing out your project idea. For this step, please write a proposal based on the project idea you agreed on with your mentor.

Please note that you will need approval from your Mentor to move on to the next step.

**Project Instructions:**

For this step, you'll write a proposal for the site you want to build. This will help your mentor better understand your chosen capstone project idea.

This proposal should be a 1-2 page document that answers the following questions:

1. What tech stack will you use for your final project?
2. Is the front-end UI or the back-end going to be the focus of your project? Or are you going to make an evenly focused full-stack application?
3. Will this be a website? A mobile app? Something else?
4. What goal will your project be designed to achieve?
5. What kind of users will visit your app? In other words, what is the demographic of your users?
6. What data do you plan on using? How are you planning on collecting your data? You may have not picked your actual API yet, which is fine, just outline what kind of data you would like it to contain. You are welcome to create your own API and populate it with data.

In brief, outline your approach to creating your project (knowing that you may not know everything in advance and that these details might change later).
Answer questions like the ones below, but feel free to add more information

- What does your database schema look like?
- What kinds of issues might you run into with your API? This is especially important if you are creating your own API, web scraping produces notoriously messy data.
- Is there any sensitive information you need to secure?
- What functionality will your app include?

- What will the user flow look like?
- What features make your site more than a CRUD app? What are your stretch goals?

**Here is the template to get started.** Use this template to help get you started right away. Once the proposal is complete, please let your mentor know that this is ready to be reviewed.

| Type | Description | Fill in |
|---|---|---|
| Stack Focus | Is the front-end UI or the back-end going to be the focus of your project? Or are you going to make an evenly focused full-stack application? | |
| Type | Will this be a website? A mobile app? Something else? | |
| Goal | What goal will your project be designed to achieve? | |
| Users | What kind of users will visit your app? In other words, what is the demographic of your users? | |
| Data | What data do you plan on using? How are you planning on collecting your data? You may have not picked your actual API yet, which is fine, just outline what kind of data you would like it to contain. You are welcome to create your own API and populate it with data. | |

## Breaking down your project

When planning your project, break down your project into smaller tasks, knowing that you may not know everything in advance and that these details might change later. Some common tasks might include:

- Determining the database schema
- Sourcing your data
- Determining user flow(s)
- Setting up the backend and database
- Setting up the frontend
- What functionality will your app include?
    - User login and sign up
    - Uploading a user profile picture.

Here are a few examples to get you started with. During the proposal stage, you just need to create the tasks. Description and details can be edited at a later time. In addition, more tasks can be added in at a later time.

| Task Name | Description | Example |
|---|---|---|
| Design Database schema | Determine the models and database schema required for your project. | Link |
| Source Your Data | Determine where your data will come from. You may choose to use an existing API or create your own. | Link |
| User Flows | Determine user flow(s) - think about what you want a user's experience to be like as they navigate your site. | Link |
| Set up backend and database | Configure the environmental variables on your framework of choice for development and set up a database. | Link |

| | | |
|---|---|---|
| Set up frontend | Set up frontend framework of choice and link it to the backend with a simple API call for example. | Link |
| User Authentication | Fullstack feature - ability to authenticate (login and sign up) as a user | Link |

## Labeling

Labeling is a great way to separate out your tasks and to track progress. Here's an example of a list of issues that have labels associated.

| Label Type | Description | Example Label |
|---|---|---|
| Difficulty | Estimating the difficulty level will be helpful to determine if the project is unique and ready to be showcased as part of your portfolio - having a mix of task difficulties will be essential. | Easy, Medium, Hard |
| Type | If a frontend/backend task is large at scale (for example: more than 100 additional lines or changes), it might be a good idea to separate these tasks out into their own individual task. If a feature is smaller at scale (not more than 10 files changed), labeling it as fullstack would be suitable to review all at once. | Frontend, Backend, Fullstack |
| Stretch Goals | You can also label certain tasks as stretch goals - as a nice to have, but not mandatory for completing this project. | Must Have, Stretch Goal |

Please create a GitHub repository for this Capstone Project and label it accordingly. All code and further documentation associated with this project should be added to this repository.

Once you've added your proposal, please submit a link to your repository for your mentor to review. You may have to iterate on this submission several times before it is approved

**NOTE**: **You will need a mentor's approval before moving on to the next step of the Capstone.**

# **Project Proposal**: Homeopathy/Alternative Remedy and Education App

## Tech Stack

- Frontend (Mobile & Web):
  - React Native: For developing a cross-platform mobile application that runs on iOS and Android using a single codebase.
  - React: For building the web application, ensuring a consistent and interactive user experience across platforms.
- This may change after I complete the curriculum on the backend
- Backend:
  - Node.js with Express: For handling API requests, server-side logic, and integration with the database.
- Database:
  - MongoDB: A NoSQL database to store and manage static content and any additional data needed by the application.
- Content Management:
  - Strapi or Contentful: Headless CMS to manage and deliver static content through an API.
- Hosting:
  - Heroku: For deploying and scaling the backend services.
  - Vercel or Netlify: For hosting the web frontend.

## Focus of the Project

The project will be an evenly focused full-stack application, integrating both frontend and backend components. The aim is to create a seamless and cohesive user experience across mobile and web platforms, ensuring that users can access educational content easily and intuitively.

## Project Type

- Mobile App: Cross-platform application for iOS and Android.
- Website: A web application accessible through modern browsers.

## Project Goal

The goal of the project is to provide accessible, evidence-based educational content for Homeopathic/Alternative remedies for common ailments. The app will serve as a reliable resource for people looking for a natural alternative to BigPharma and conventional solutions to ailments, offering valuable information and guidance on this subject.

## User Demographic

The primary users of the app will be:

- Patients looking for a nature based solution to common ailments.
- Patients that want to address the problem instead of treating the symptom.
- Healthcare Professionals: Healthcare providers seeking a reliable resource as a natural alternative.

## Data and API

- Data: The content will include lists of ailments with remedies, lists of remedies with ingredients, dosage, side effects, and tips and articles about Homeopathic remedies.
- Data Collection: Content will be managed using a headless CMS (Strapi or Contentful) and delivered via an API to both the mobile app and website.
- API: The API will provide endpoints for retrieving content, including articles, tips, and multimedia resources. The API may be custom-built or integrated with the CMS API.

## Project Approach

1. Database Schema
    - Content: Stores articles, tips, and multimedia resources with fields such as title, body, category, and media URLs.
    - Categories: Stores categories for organizing content (e.g., Homeopathic remedies, Alternative remedies, Ailments, Where to access).
2. Potential API Issues
    - Data Consistency: Ensuring content is consistently delivered across platforms.
    - Scalability: Handling increasing amounts of data and user requests efficiently.
    - Error Handling: Managing and reporting API errors to ensure a smooth user experience.
3. Sensitive Information
    - No sensitive or personal data will be collected or handled. The focus is on delivering static educational content.
4. Functionality
    - Content Display: Users can view educational articles, tips, and multimedia resources.
    - Search and Filter: Users can search for and filter content based on categories and keywords.
5. User Flow

- Homepage: Displays featured content and navigation options.
- Content Pages: Users can access detailed articles and resources. Includes: Remedies (Browse by Ailment list, browse by Remedy list, Search, Learn.
- Search/Filter: Users can search for specific topics and filter content by category.

6. Stretch Goals
    - User Feedback: Incorporate a feedback system to gather user insights and improve content.
    - Interactive Features: Add features like quizzes or interactive tools to enhance user engagement.
    - Multi-language Support: Expand content accessibility by offering multiple language options.