# Problem statement

People look forward to vacations and cannot wait to start it, but before it starts, they need to compile a plan. However, compiling a vacation plan requires making multiple search queries. For example, what type of transportation is commonly used in the area, which accommodation should be booked throughout the vacation, which food spots are found in the vacation destination, as well as what attractions would be available. This leads to multiple browser tabs open and hours spent on compiling the plan.

Additionally, a budget needs to be calculated for a vacation plan. This might require the conversion of prices from a destination’s currency to the user’s preferred currency. To do so, the exchange rate needs to be known, and calculations need to be executed. This further adds to the hours spent on compiling the plan.

# Proposed solution

VacayPay solves this by doing the following. Based on the selected vacation destination, as well as their selected spender preferences, the application will retrieve a selection of transportation, accommodation, food spots, and attraction for that destination. This allows a user to select from the retrieved items. After the selection, the application will provide a preview detailing the estimated budget for the vacation. Once the user is satisfied, the vacation plan is saved and can be viewed, edited or deleted at a later stage.

# Functional requirements

Users can:

* Create, update, or delete their account
* Add, view, update, or delete their budget plans
* View budget plans based on preferred currency

# Non-functional requirements

The application will use JSON Web Tokens (JWTs) and it will hash the user’s passwords before they are stored in the database for enhanced security. It will also be a responsive application to enable accessibility for desktop, Android and iOS devices.

# Feasibility study

## Market feasibility

There are several vacation planner apps with budgeting features. For example, Stippl offers itinerary planning, travel budgeting, and an AI feature that generates tailored itineraries based on your destination. It also helps find accommodation, transportation, restaurants, and activities. Its budget planner lets users set limits, track purchases, and split expenses among co-travellers.

My solution is similar in that it compiles a list of accommodations, food spots, transportation options, and attractions to visit. However, unlike existing planners, it uniquely provides users with an estimated total expense for their entire vacation before the trip begins—a feature not offered by current apps like Stippl. This budgeting capability is especially helpful for individuals visiting unfamiliar places or those who want to know in advance how much they should save. As a result, users can focus on enjoying their trip without the need to track each individual expense during their vacation.

This web application is best suited for individual’s above the age of 18 who have the means to fund their own vacations and require an estimated budget for their upcoming vacations.

## Technical feasibility

The proposed solution will make user of ReactJS, CSS, HTML, and JavaScript for the development of the frontend, MySQL and SQL for the database, and ASP.NET Core Web API to allow the frontend to make requests to the backend. The solution will be a web-application to make it accessible to desktop, Android and iOS devices.

## Financial feasibility

By complexity, the ReactJS web application needs to make http requests to an ASP.NET Core web API to add, update, retrieve or delete data in the MySQL database. The frontend development costs R90 000, the backend development costs R120 000, and the Database & APIs costs R170 000. Finally, testing and quality assurance costs R 60 000. Therefore, the total cost of developing this web application is R440 000.

## Schedule feasibility

The project scope:

* Landing page
* Home page
* Vacation plan page (will be used for adding or editing the plan)
* View vacation plan page (will include editing and deleting options)
* Profile page

Project estimated effort:

* Write documentation detailing problem, solution, use cases, requirements (2 – 5 day)
* Design Entity Relational Database Diagram (5 – 7 days)
* Develop Backend with test data (7 – 10 days)
* Develop ASP.NET Core web API (5 - 10 days)
* Develop frontend (10 - 15 days)

|  |  |
| --- | --- |
| Task | Deliverable deadline |
| Write documentation | 24 September - 28 September 2025 |
| Design ERD Diagram | 28 September - 4 October 2025 |
| Develop Backend | 4 October - 12 October 2025 |
| Develop API | 12 October - 19 October 2025 |
| Develop frontend | 19 October - 26 October 2025 |