CSCI 5709

Assignment 1

Individual Submission

Name: Benny Daniel Tharigopala

Banner ID: B00899629

Instructor: Shehzeen Huda

Teaching Assistant: Srikrishna Sasidharan

GitLab URL: CSCI_5709_ASSIGNMENTS_B00899629_BENNY

Heroku App URL: https://csci-5709-a1-benny-tharigopala.herokuapp.com/

Feature and Task Details

The first feature perceived in my team's application "Expense Tracker", relevant to Assignment -1, is the **In-App Payment** module. This module has four tasks, namely:

1. Add a Payment Method
2. Initiate a Payment
3. View Payment Status
4. Analyze Payment History

The task chosen for this assignment focuses on the facilitation of adding and storing payment methods, that is, credit cards for present and future utilization. Identical to real world credit card data storage data input interfaces, I have developed a user interface through which "Payers", or any other User Personas can store credit card information, which they can readily access and use to make payments. The interface obtains standard card attributes such as Card Number, Expiry Month, Year, CVV and Name on Card. Once stored, "Payers" can view their cards on their "Checkout" page before they initiate payments to a "Payee". Several E-commerce websites' pages were analyzed as part of this Assignment, to conceptualize and subsequently render an elegant and simple interface.

Design Choices and Justifications

Every project has diverse applications and user requirements. As such, this project too requires a simple, and intuitive design for users that, is visually appealing, promotes accessibility, time-optimization, and positive user experience. My team and I opted for ReactJs as our UI development library due to its popularity, features, and moderate learning curve. ReactJS is declarative, therefore, we can offer more services with relatively lesser effort as compared to other libraries or frameworks such as Angular. The improved performance obtained through a Virtual Dom and the capability to reuse components **[2]** was one of the major reasons for our decision to render user interfaces with ReactJS. As such, a library "**react-credit-cards [4]"** was used to render sassy and professional looking webpages for adding and displaying credit cards in the application. The information retrieved from users is displayed on the card and an exciting feature in this library is the capability to turn the image of a "credit card" on the "Payment

Methods" webpage, to view the Card Verification Value (CVV). Currently, I have planned to integrate the **Stripe API** [5] with the Payment interface, to accept and send payments. Stripe doesn't require Users to have an account thereby making payments seamless and rapid. In addition, Stripe offers SSL and encryption features [6] therefore Payment security is one less issue we have to be worried about. The current color scheme is predominantly blue, for an aesthetic appearance, but it will be updated in the future when the team has agreed upon an appropriate color palette and design scheme for our application. Currently, I have also utilized Tailwind CSS

User Experience and Task Flow

Figures 1-4 [3] represent the path taken by Users while using the Payment Interface. The corresponding Use Cases for these Task Flow diagrams can be found between Pages [25-29] of the Group Submission document.

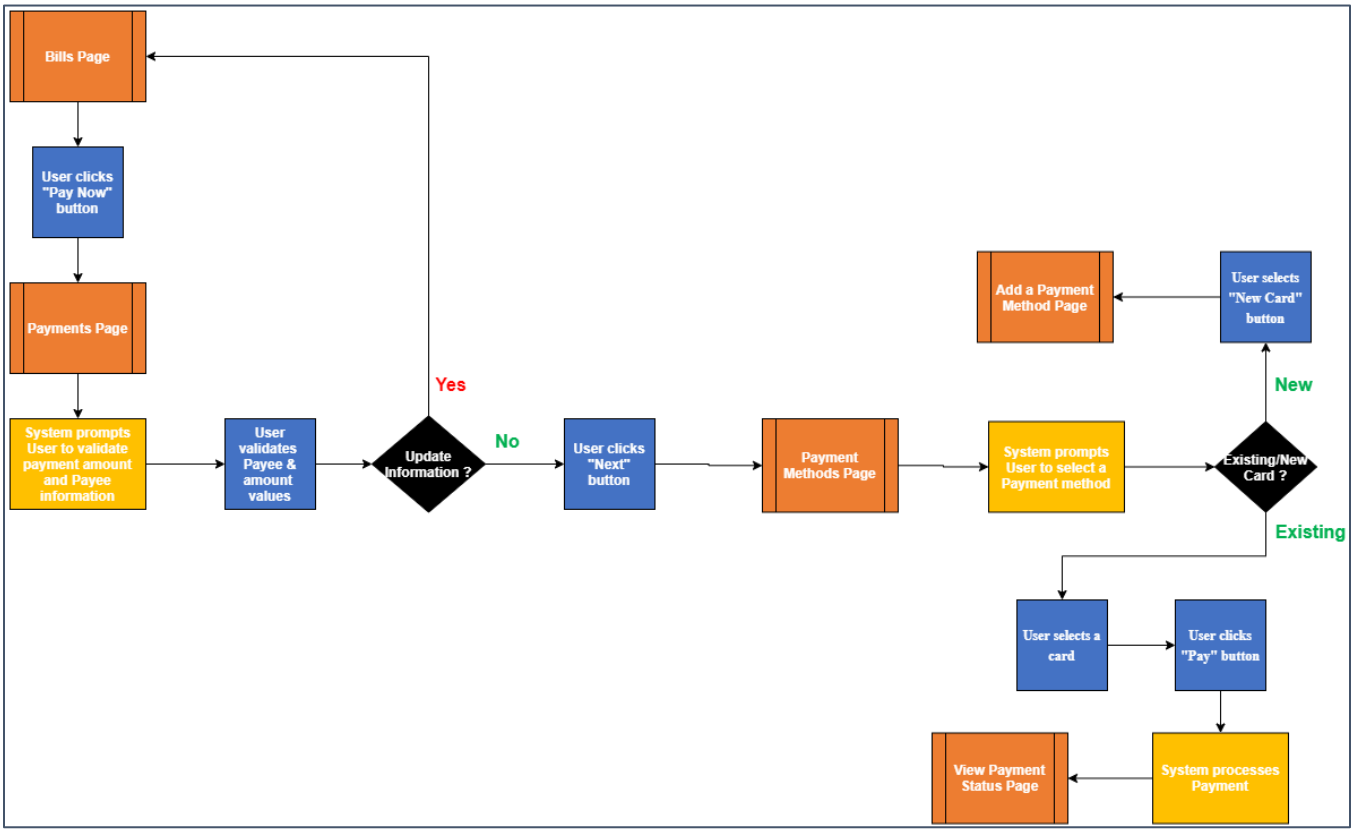

Figure 1 1: Initiate a Payment – Task Flow

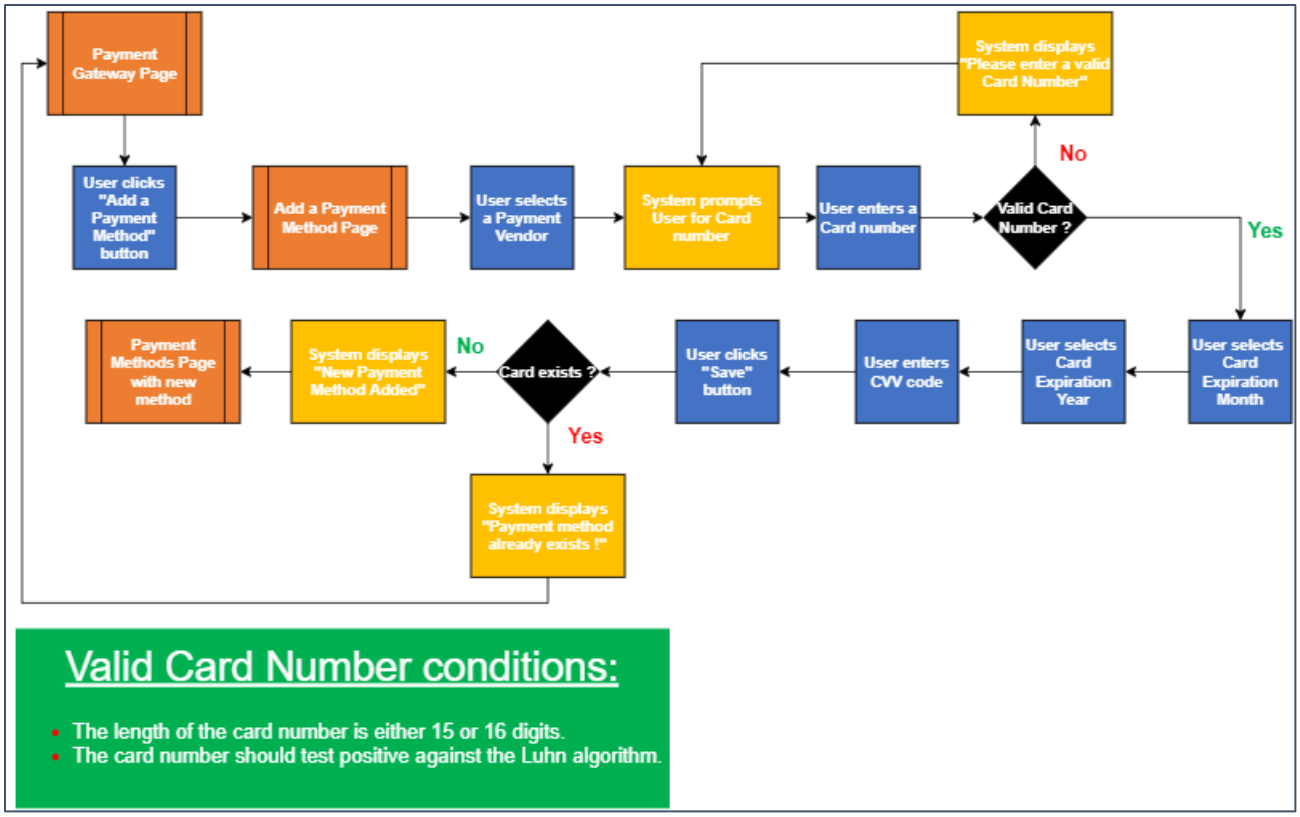

Figure 22: Add a Payment Method – Task Flow

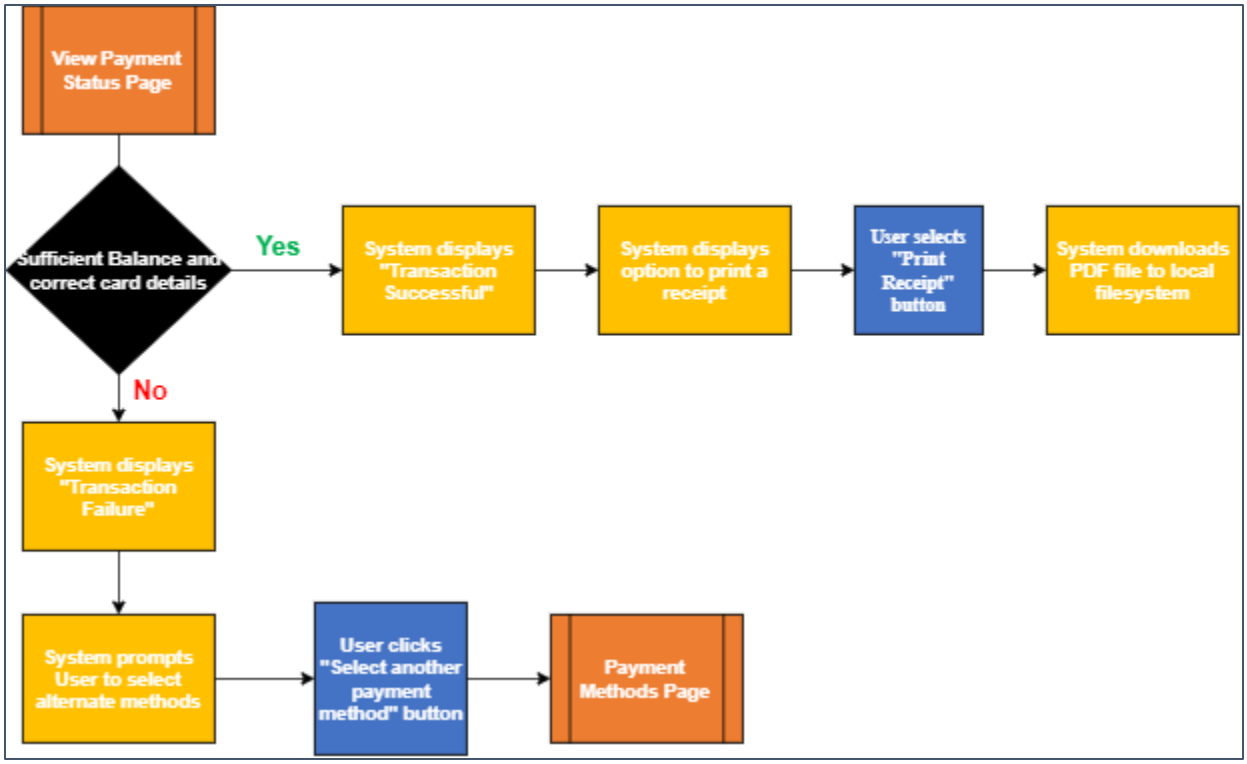

Figure 33: View Payment Status – Task Flow

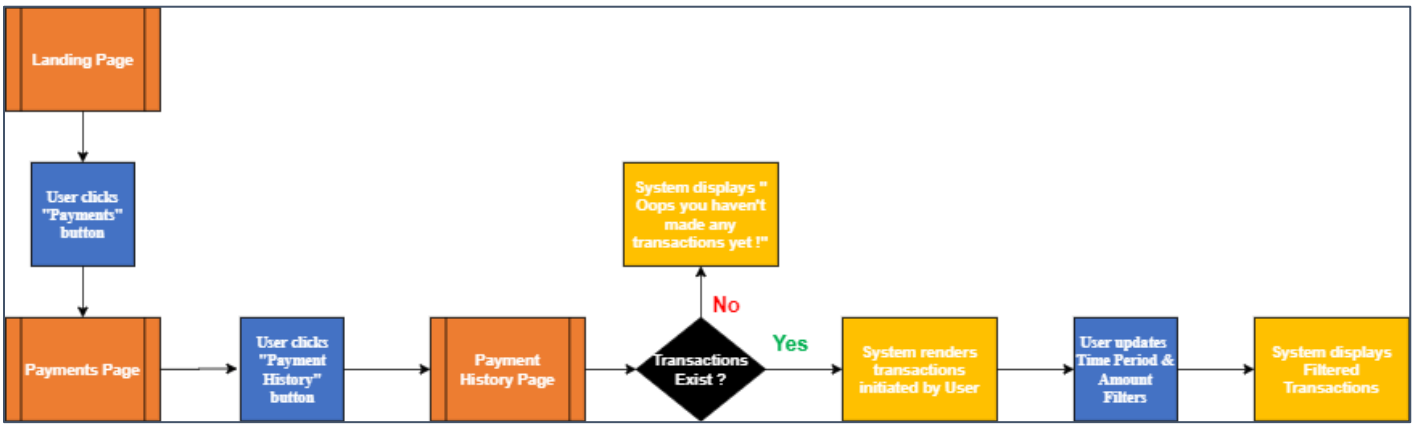

Figure 44: View Payment History – Task Flow

Lo-Fidelity Prototype

Figures 5-7 **[1]** represent the webpages I conceptualized for the In-App Payment feature. After initiating a payment, the user is routed to a "Payment Methods" page where they can add, remove, edit, and select a card to proceed with a transaction. When the User has not stored any card details in the System, the page notifies the User that they have to add a credit card to the system, in order to proceed with the payment. Otherwise, the "cards" that were previously added by Users, are displayed. The "Add a Payment Method" form obtains the details of a credit card from Users and the "Payment Methods" page then renders the data in the form of a credit card.

In-App Payment Module: Add a Payment Method Wireframes:

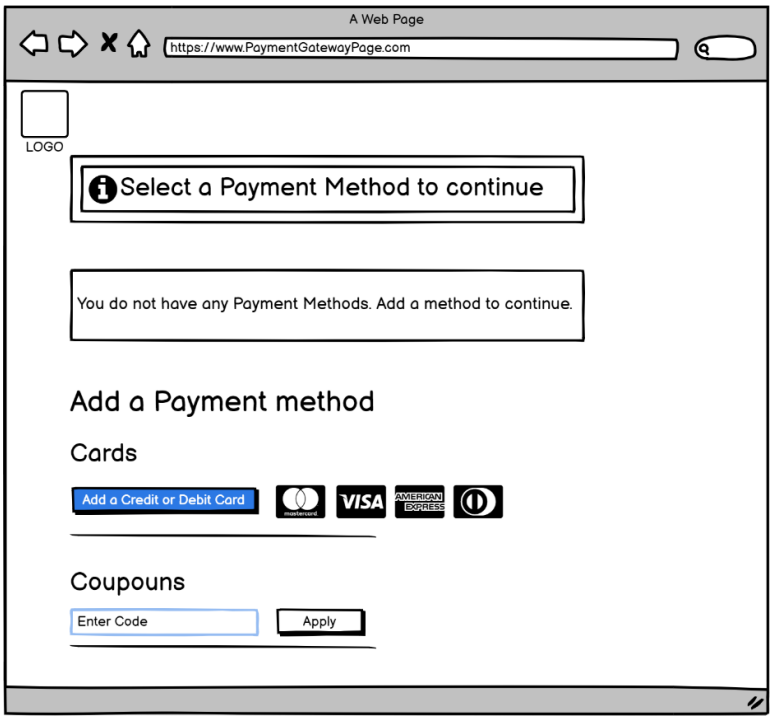

Figure 55: Add a Payment Method: No Cards Exist – Wireframe

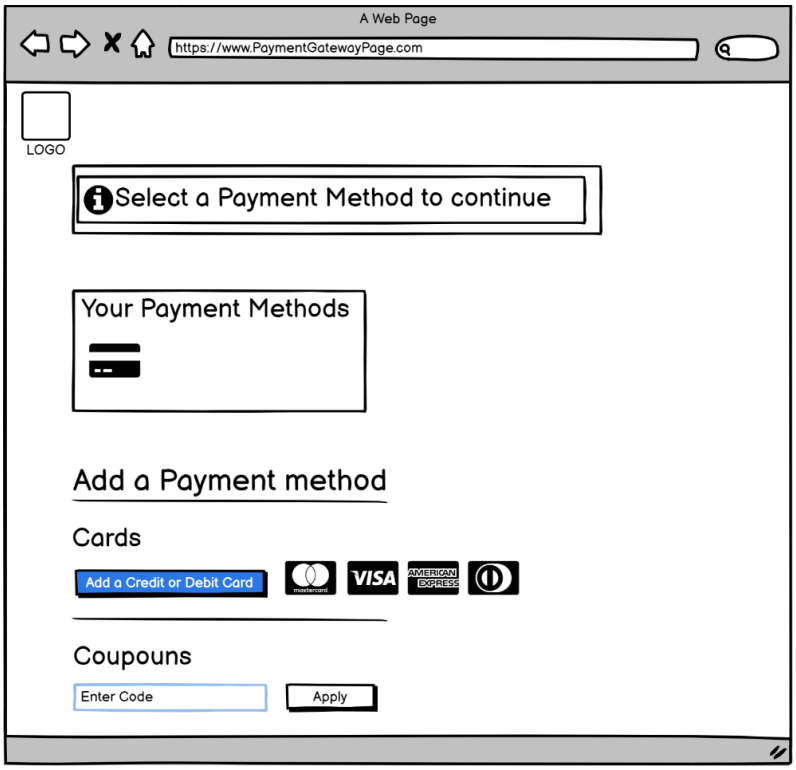

Figure 66: Add a Payment Method: Cards Exist – Wireframe

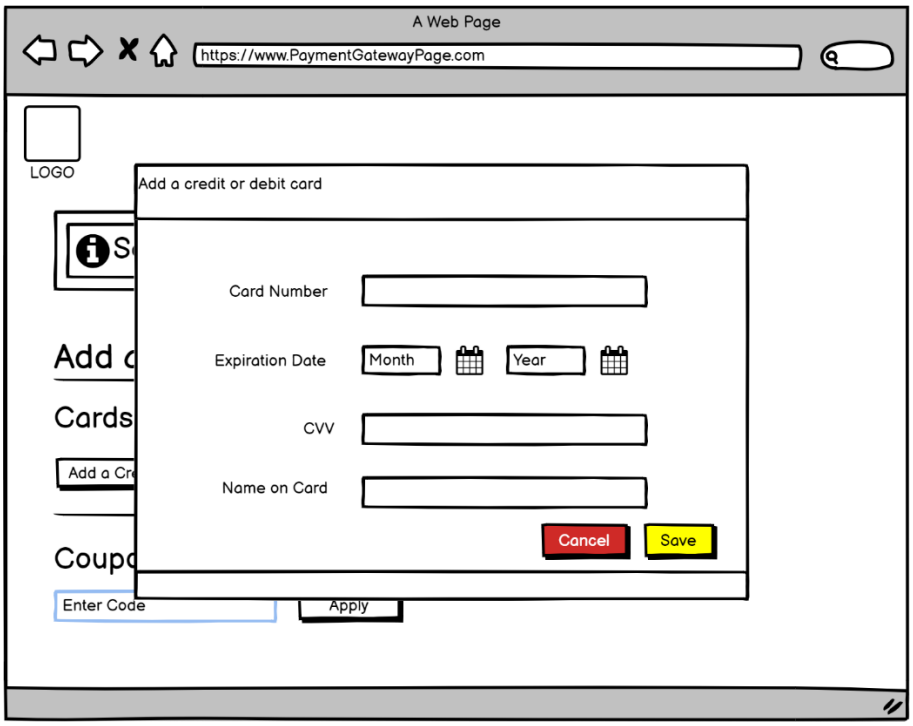

Figure 77: Add a Payment Method: Input Details – Wireframe

Citations

[1] "Balsamiq Cloud", *Balsamiq.cloud*, 2022. [Online]. Available: https://balsamiq.cloud/. [Accessed: 04- Jun- 2022]

[2] Deshpande, Chinmayee. "What Is ReactJS: Introduction to React and Its Features." *Simplilearn.com*, 30 May 2020, www.simplilearn.com/tutorials/reactjs-tutorial/what-is-reactjs. Accessed 24 May 2022.

[3] "Flowchart Maker & Online Diagram Software", *App.diagrams.net*, 2022. [Online]. Available:https://app.diagrams.net/. [Accessed: 04- Jun- 2022]

[4] Cardoso, Casio. "React-Credit-Cards." Npm, 12 Nov. 2020, www.npmjs.com/package/react-credit-cards. Accessed 5 June 2022.

[5] "Stripe API Reference." Stripe.com, 11 Aug. 2017, stripe.com/docs/api. Accessed 26 May 2022.

[6] Ozanne, Béatrice. "Statrys • How to Connect Stripe to Your Website." Statrys.com, 5 Nov. 2021, statrys.com/blog/connect-stripe. Accessed 27 May 2022.