Project Proposal - BudgetGator

**TEAM NAME**

(EA)^2

**TEAM ROLES**

*Product Owner*: Anushka Kapoor

*ScrumMaster:* Erin Foege

*Development Team*: Anushka Kapoor, August Williams, Erin Foege, Emma Baumgartner

**PROJECT MANAGEMENT TOOL**

Jira

**LANGUAGES, FRAMEWORK, TECHNOLOGIES**

Javascript

[node.js](http://node.js)

express.js

React Native

Plaid API

mongoDB

**PROPOSED SOLUTION**

* Manage multiple credit cards
* Help with budgeting and managing expenses
* Have a pie chart (and other visualizations) showing expenses by categories (including grocery stores, dining, entertainment, recreation, rent, etc.)
* Allow for the manual entry of expenses
* Analyzes spending to show abnormal spending patterns or over budget categories
* Users can input targeted money goals and then help the user stay on budget
* Allow students and other users to invite parents/benefactor to the view the account

We aim to create a platform that allows users to manage their finances by providing an all-encompassing app to help with budgeting and expense management. This app will provide the most benefit to people who have multiple cards and/or expenses coming from different accounts, as this app will centralize spending insights. The app will allow for users to link multiple credit cards to the app through a secure platform, and they can then be able to see their spending broken down into categories (like dining, entertainment, rent, etc.). Spending by category will then be displayed in visuals like pie charts. The app will analyze the user’s spending and break down abnormal spending patterns while bringing attention to over budget categories. The user will also be able to manually enter expenses coming from an unlinked account to be viewed alongside their other spending. Users can input targeted money goals in order to get personalized suggestions and advice for sticking to their budget. Another helpful feature will be that a student or other user can invite benefactors to view the account and monitor their spending.

* Implements user authentication with at least two distinct user types (e.g., administrator and member)
* Implements a dynamic database of user and item data
* Implements one or more “operations” involving some form of frontend-backend-database communication

**PROJECT VISION**

For young adults who want to better manage their finances, BudgetGator is a financial application that allows the user to link their credit cards and get a comprehensive analysis on their spending habits.

**RISK MANAGEMENT PLAN**

| **Explanation** | **Probability** | **Effects** |
| --- | --- | --- |
| Member(s) of the team fall ill during critical times for the project | Moderate | Serious |
| The time needed to complete sprints is underestimated | High | Serious |
| The chosen software tools can’t be integrated properly | Moderate | Tolerable |
| Bug fixes and defect repairs take extended amounts of time | Moderate | Serious |
| Team member training and research into languages, framework, and technology is insubstantial | High | Insignificant |

Our avoidance strategies are proposed as:

1. Create a deadline calendar where deadlines are a few days earlier than actually needed, so that if something happens to any individual member of the team, there is still built in time for someone else to finish the work. In other words, we will implement soft deadlines and hard deadlines.
2. Ensuring the work is divided and allocated efficiently so everyone can work specifically on reasonable parts of the project that can be completed in the time of the sprint.
3. If mongoDB is not working, we can also use PostgresSQL. In other words, have backup technologies that we can use.

Our minimization strategies are proposed as:

1. To mitigate the effects of bugs and defects, we will test extensively and not push changes to the main branch that have not been tested separately. We will practice version control, so we can fall back to a previous working version if necessary

In the case a risk occurs despite efforts of avoidance and minimization, our contingency plan is:

* If a team member falls ill, we will ensure the project continues smoothly by splitting up that person’s tasks and responsibilities. Documentation of what that team member was working on, and how they have currently implemented/plan to go about implementing that functionality will be integral in ensuring work can be picked up without much difficulty and need for research.
* If we end up going overtime with a sprint, we will ensure the project is still completed on time by considering the cutting of app functionality that is useful but not integral to the solution the project aims to resolve. We will do this by understanding the order we should implement functionality, as well as understanding the ordering of importance of functionalities.
* If the chosen software tools can not be used, we have made sure there are other substitute tools that can be used. Dummy data may be used to ensure certain functionality before the implementation of real data.
* If bug fixes and defect repair prove difficult and time consuming, we will ensure the project continues smoothly with more thoughtful automated testing. Documentation and multiple versions will allow us to roll back to a version without the bug.