

iBudget

Software Requirements Specification

Approvals: L. Assayah LA date: 2/27/12
C. Leung CL date: 2/27/12
Q. Pham QP date: 2/27/12
V. Velev VV date: 2/27/12
J. Reimels JR date: 2/27/12
V. Dineva VD date: 2/27/12

Revision History

Date	Author	Version	Reason
2/27/12	V.Velev L. Assayah Q. Pham V. Dineva C. Leung J. Reimels	1.0	First Draft
3.26/12	V.Velev C. Leung	2.0	Modified section 2.1 Modified use case diagram Added Source Class Diagram Added Object Class Diagram

Table of Contents

1. Introduction	3
1.1. Purpose	3
1.2. Scope	3
1.3. References	3
2. Overall Description	3
2.1. Product Perspective	3
2.2. Product Functions	4
2.3. User characteristics	7
2.4. Constraints	7
2.5. Assumptions and dependencies	7
3. Specific Requirements	7
3.1. Functional Requirements	7
3.2. Non-functional requirements	15

1. Introduction

1.1. Purpose

This document provides all of the requirements for the *iBudget* project.

1.2. Scope

This document covers the design for release 1.0 of *iBudget*.

1.3. References

IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998, Revision of IEEE Std 830-1993)

2. Overall Description

iBudget is to be an online application which collects, processes, and displays the user's financial information in an effective manner. *iBudget* will require the user to upload data from a financial institution in the form of a CSV file. This data will be processed by the back-end, and a report will be displayed to the user as he or she requests it. For example, the user will be able to customize the report and view it in the form of a budget statement. Furthermore, the user will be able to create custom categories and map any of the data to them. This allows for further flexibility in customizing the report of the uploaded information. In addition to these features, the application will allow the user to create and access an account.

2.1. Product Perspective

iBudget is intended to address the need for a budgeting and personal finance tool that does not require a secure connection to a financial institution or one's login credentials to that institution. It is targeting a market that is conscious of internet safety and aware of the risks associated with providing sensitive information to an online service. *iBudget's* appeal is that it will provide peace of mind to its clients without sacrificing any of the functionality offered by its competitors.

An example of related work is www.mint.com – a personal finance and budgeting tool. Mint's advantage is that it provides a complete and easy to use framework for easily managing and monitoring one's finances. Its user-friendliness is its main asset. It also, however, requires its users to provide information such as bank account credentials. This discourages many from registering for the service.

2.2. Product Functions

The *iBudget* application has two active actors. The User and the Administrator access *iBudget* through the Internet. A Use Case diagram of the application is shown in Figure 1.

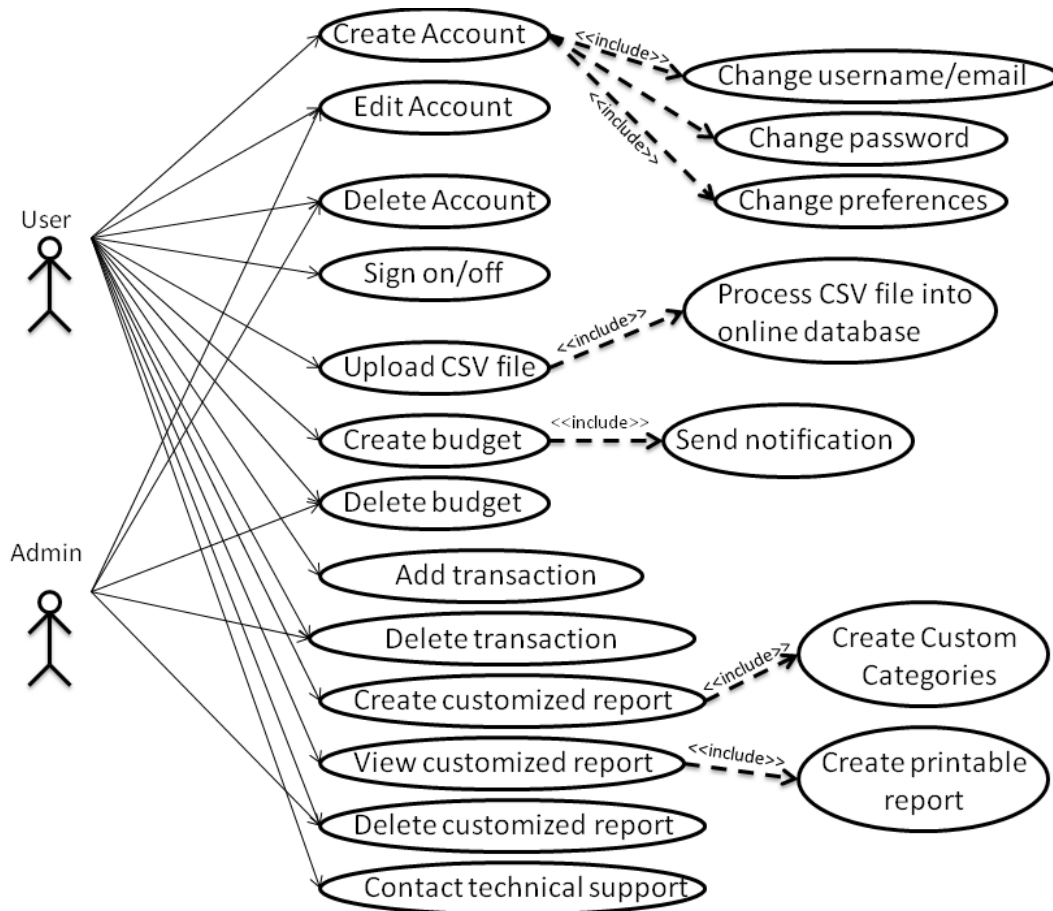


Figure 1: iBudget Use Case Diagram

2.2.1. “Create Account” User Case

Actor: User

Brief Description:

The User selects to *Create Account*, and the system presents a form with information to be filled out by the User.

2.2.2. “Edit Account” Use Case

Actor: User, Admin

Brief Description:

The User selects to modify account information and the system displays the modifiable information with an option to resubmit the new information. The Admin also has permission to edit one's account.

2.2.3. "Delete Account" Use Case

Actor: User, Admin

Brief Description:

The User selects to delete his or her account and system removes the information from the database. The Admin also delete a User's account.

2.2.4. "Sign on/off" Use Case

Actor: User

Brief Description:

The User enters his login information to *Sign On*, and the system presents the User's homepage. The User selects to *Sign Off* and the system presents the homepage of the website.

2.2.5. "Upload CSV File" Use Case

Actor: User

Brief Description:

The User uploads a CSV file that gets processed by the system.

2.2.6. "Create Budget" Use Case

Actor: User

Brief Description:

The User has the option to create a budget based on the CSV file that has been uploaded. The system will process the data and output the desired budget.

2.2.7. "Delete Budget" Use Case

Actor: User, Admin

Brief Description:

The User and Admin select *Delete Budget*, and the system removes the user specified budget configuration.

2.2.8. “Add Transaction” Use Case

Actor: User

Brief Description:

The User has the option to add a single transaction that will be processed by the system.

2.2.9. “Delete Transaction” Use Case

Actor: User, Admin

Brief Description:

The User and Admin select *Delete Transaction*, and the system removes existing transactions from the User’s account.

2.2.10. “Create Customized Report” Use Case

Actor: User

Brief Description:

The User creates a customized report from the existing data in his account, and the system processes the data to create the requested report.

2.2.11. “View Customized Report” Use Case

Actor: User

Brief Description:

The User selects *View Report*, and the system displays the selected report.

2.2.12. “Delete Customized Report” Use Case

Actor: User, Admin

Brief Description:

The User and Admin select *Delete Report*, and the system removes the selected report.

2.2.13. “Contact Technical Support” Use Case

Actor: User

Brief Description:

The User can contact the technical support team of *iBudget*, and the system will process the request and forward the message.

2.3. Source Class Diagram

Refer to Appendix.

2.4. Source Class Diagram

Refer to Appendix.

2.5. User characteristics

iBudget's target demographic is working adults conscious of internet safety and aware of the risks associated with providing sensitive information to an online service.

2.6. Constraints

None.

2.7. Assumptions and dependencies

The team is assuming that the user will be using the latest version of one of the four major desktop browsers: Internet Explorer, Mozilla Firefox, Google Chrome, and Safari.

3. Specific Requirements

3.1. Functional Requirements

3.1.1 Create an account

The user has to first sign up to get an account and be able to log in the next time he will access the website.

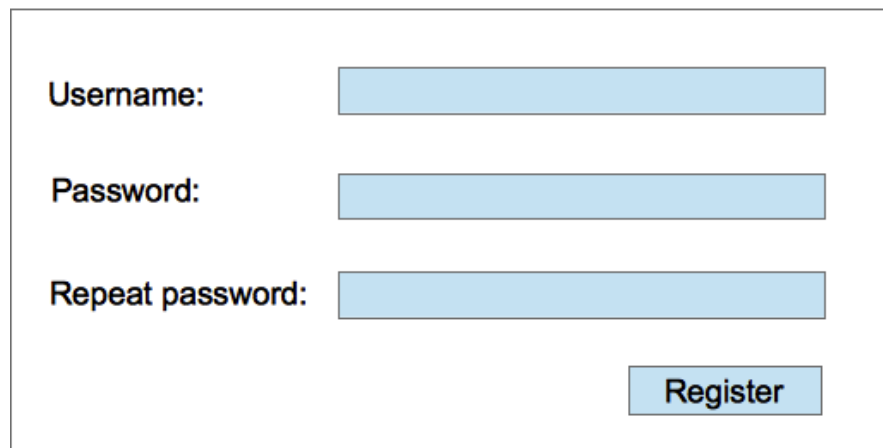
Actions:

- Click on the “Register” hyperlink. The user is then brought to the corresponding webpage.
- Fill a form with the fields:
 - Username (must be at least 3 characters and unique)
 - Password (must be at least 8 characters, include at least 1 upper case character, 1 lower case character, 1 number and 1 symbol)
 - Password Confirmation
- Click on the “Register” button

Results:

- If the user failed to fill the form correctly, the form is displayed again with the reason for the error in the corresponding field (not enough characters, username already used...)
- If the user succeeded then a message saying “Your account has been created” is displayed. The user is then redirected to the homepage and directly logged in.

Create an account



The form is titled "Create an account" and is enclosed in a light gray border. It contains three text input fields, each preceded by a label: "Username:", "Password:", and "Repeat password:". The input fields are light blue with a thin gray border. At the bottom right of the form is a blue button with the text "Register" in white.

3.1.2 Login

The user has to login to use the functionalities the website.

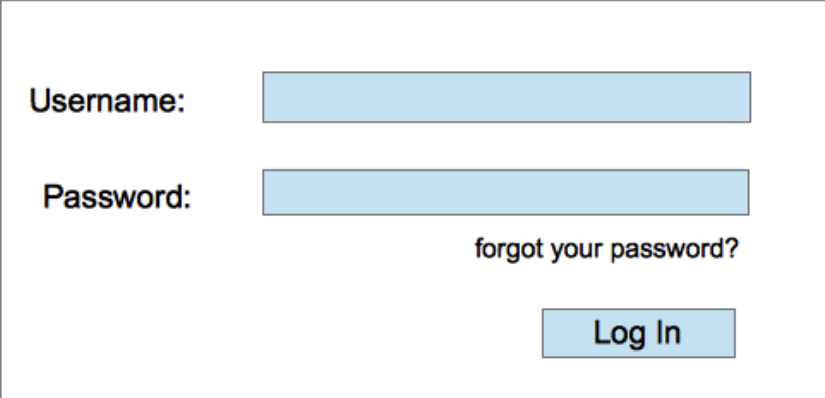
Actions:

- The user must click on the “log in” hyperlink
- The user is taken to the log in page where he has to fill out the username and password fields
- Then he must click on the “Log In” button

Results:

- If the user failed to log in, the window is still displayed and an error message appears at the top of the form saying “Wrong username or password”
- If the user succeeded, the window disappears and the user is logged in in the homepage.

Log In to iBudget



A login form titled "Log In to iBudget". It contains two input fields: "Username:" and "Password:". Below the password field is a link that says "forgot your password?". At the bottom right is a "Log In" button.

3.1.3 Change password

The user can change his password anytime when he is logged in.

Actions:

- The user clicks on the “Change password” hyperlink.
- He has to fill the fields:
 - Old Password
 - New Password (must be at least 8 characters, include at least 1 upper case character, 1 lower case character, 1 number and 1 symbol.)
 - Confirm new password
- He clicks on the “Ok” button

Results:

- If the user failed to fill the form correctly, the form is displayed again with the reason for the error in the corresponding field (not enough characters)
- If the user succeeded then a message saying “Your password has been changed” is displayed. The user is then redirected to the homepage.

3.1.4 Get a new password if the user forgets it

Sometimes the user can forget his password. The application can provide a way to get a new password quickly.

Actions:

- The user clicks on the “forgot password” hyperlink.
- A message saying if the user wants to have a new password is displayed. If the user clicks on the “Yes” button, an email is sent to the user’s email address.
- The user has to open the email and click on the link.
- He is redirected to a webpage that allows him to type a new password in a text field. He has to type the password twice to confirm it.
- He clicks on the “Ok” button. A confirmation email is then sent to the user telling him that his password has indeed been changed.

Results:

- The user has a new password and so he can log in in the website with the new password and can access his data without any difficulties.

3.1.5 Log out

The user can log out of the application before leaving the website or if he wants to log in with another username.

Actions:

- In every page of the website, the user can click on the “Log out” hyperlink

Result:

- The user is logged out and redirected to the homepage of the website.

3.1.6 Upload CSV files

The user can upload his bank files in CSV format in order to visualize them the way he wants afterwards.

Actions:

- The user clicks on the “Upload CSV file” hyperlink on the navigation bar. The corresponding webpage is then displayed.
- The user clicks on the “Upload file” button.
- The user can then choose the file he wants to upload on his computer and then clicks “OK”

Results:

- If the user failed, (wrong format, file can’t be read...), an error message is displayed and the user is invited to try again with uploading the file or another file.
- If the user succeeded, the data has been successfully uploaded and the user is asked if he wants to upload another file or not. If not, he is redirected to the homepage.

3.1.7 Edit User profile

The user can access his profile to see his information and can edit ~~them~~ it at will.

Actions:

- The user clicks on the “Profile” hyperlink in the navigation bar.
- The webpage is displayed and all the fields that the user can modify are organized in editable text fields. The user can modify information like city, zip code, state, email address etc...
- When the user is done editing, he clicks on the “Save” button.
- If he doesn’t want to edit after all, he can click on the “Cancel” button

Results:

- If the user clicked on “Save” and if there is no errors, he is redirected to the homepage and his profile has successfully been edited.
- If the user clicked on “Cancel”, A confirmation message is displayed in a pop-up window: “Are you sure you want to cancel?” If the user clicks “Yes”, he is redirected to the homepage and his information has not been changed. If he clicks on “No”, the window disappears and he can continue editing.

3.1.8 Create custom categories

The custom categories enable the user to associate data the way he wants and not be limited by the default categories.

Actions:

- The user clicks on the “Create Category” hyperlink on any page of the website.
- The webpage is displayed and the user can fill the text field with the name he chose.
- If he wants to add several names at a time, he can click on the “+” button. Another text field will appear below every time he clicks on the button.
- When the user is done with the names, he clicks on the “Save” button.

Results:

- The categories are saved and can be used later when the user wants to map data with categories

3.1.9 Create custom financial institute mapping for data import

If user cannot map the imported CSV with any of the proposed mappings, he can create a customize mapping.

Actions:

- The user clicks on the “Create Custom Mapping” hyperlink on any page of the website.
- A list of all users’ imported CSV data layouts is displayed then the user can select one of layouts then hit the NEXT button.
- All fields of the selected layout are displayed in the first column. The second column shows combo boxes where the user can choose one field from the iBudget’s predefined fields.
- When the user is done, he can click on the SAVE button to save and name the newly created mapping or he can click CANCEL to undo the changes. The application will only allow the SAVE operation when all CSV fields are mapped and mapped to unique iBudget fields.

Results:

- If the user chooses to save the mapping, the mapping is saved and can be used later to map with a similar layout.

Custom Mapping	
CSV_Field1	iBudget_Field1
CSV_Field2	iBudget_Field3
CSV_Field3	iBudget_Field5
CSV_Field4	iBudget_Field2
CSV_Field5	iBudget_Field4
<div>SAVE CANCEL</div>	

3.1.10 Set preferences

After the user has successfully created an account, he is directed to the Preferences page so that he can set his preferences such as default financial report, default chart

type, etc...

Actions:

- The webpage is displayed and all the preferences that the user can set.
- When the user is done editing, he clicks on the "Save" button. The application will only allow the SAVE operation when all preferences have been set.

Results:

- If the user clicked on "Save" and if there is no errors, he is redirected to the homepage and his preferences has successfully been set.

3.1.11 Change preferences

The user can access his preferences and change them at will.

Actions:

- The user clicks on the "User Preferences" hyperlink in the navigation bar.
- The webpage is displayed and all the preferences that the user can modify such as default financial report, default chart type, etc...
- When the user is done editing, he clicks on the "Save" button.
- If he doesn't want to edit after all, he can click on the "Cancel" button

Results:

- If the user clicked on "Save" and if there is no errors, he is redirected to the homepage and his preferences has successfully been edited.
- If the user clicked on "Cancel", A confirmation message is displayed in a pop-up window: "Are you sure you want to cancel?" If the user clicks "Yes", he is redirected to the homepage and his information has not been changed. If he clicks on "No", the window disappears and he can continue editing.

3.1.12 Delete an account

The user can delete his account from the application.

Actions:

- The user clicks on the "Profile" hyperlink on the navigation bar of the website.
- Then he has to click on the "Delete Account" button.

Results: A confirmation message is displayed

- If the user clicks on "Yes", a message saying "Your account has been successfully deleted" is displayed.
- If the user clicks on "No", the account is not deleted and the user is still on the profile webpage.

3.1.13 Edit Transaction Data

The user can edit transactional data from imported csv files.

Actions:

- The user selects a transaction
- The user selects to edit the transaction
- The user edits fields
- The user saves the edits

Results:

- The transaction is updated with the edits the user made

3.1.14 Delete Transaction Data

The user can delete any transactional data from imported csv files.

Actions:

- The user selects a transaction
- The user selects to delete the transaction
- The user selects "Yes" on the "Are you sure?" prompt

Results:

- If the user selects "Yes" on prompt:
 - The transaction is deleted
- If the user selects "No" on prompt:
 - The transaction is unchanged

3.1.15 Generate Report

The user can generate reports on spending habits.

Actions:

- The user selects "Create Report" from the home screen
- The user selects date span to report on
- The user selects type of report (Net Income, Income, or Spending)
- The user selects a sub category (By Time, or By Transaction Category)
- The user selects "Display Report"

Results:

- A Report and Graph is shown to the user
- By Time:
 - A bar graph with data shown per day
- By Transaction Category:
 - A Pie chart is shown with percentages of total for each category

3.2. Non-functional requirements

3.2.1 Consistent look for different Internet browsers.

The application should have the same look for all browsers. There can be differences related to the specific ways a browser can display some items.

We currently support the latest versions of the following browsers:

- Mozilla Firefox
- Internet Explorer
- Safari
- Google Chrome

3.2.2 Enforce strong password

When the user chooses his password, he has to respect the following:

- at least 8 characters long
- at least 1 upper case character
- at least 1 lower case character
- at least 1 number
- at least 1 symbol

3.2.3 End of session

The application should end the session if the idle lasts more than 15 minutes. The user is then logged off automatically. All the data not saved will be lost.

3.2.4 Built-in reports

The application should display any built-in report in less than 60 seconds.

3.2.5 Clarity

The application's front-end should be clear and user-friendly. There will be a navigation bar at the top of the screen for easy navigation of the site.

3.2.6 Encryption

The user's password must be single way encrypted.

3.2.7 SSL

Secure Sockets Layer (SSL) will be used to protect the privacy of all user-related data and its integrity

3.2.8 Module functionality

The Data Access layer will be abstracted from the core application code. Factory and keys and controllers will also be used to ensure modularity of the core code.

3.2.9 Cookies

The application should be able to remember logged in users by using cookies.

3.2.10 Portability

The system will be cross-platform compatible and the only requirement for users are access to the internet and a web browser.

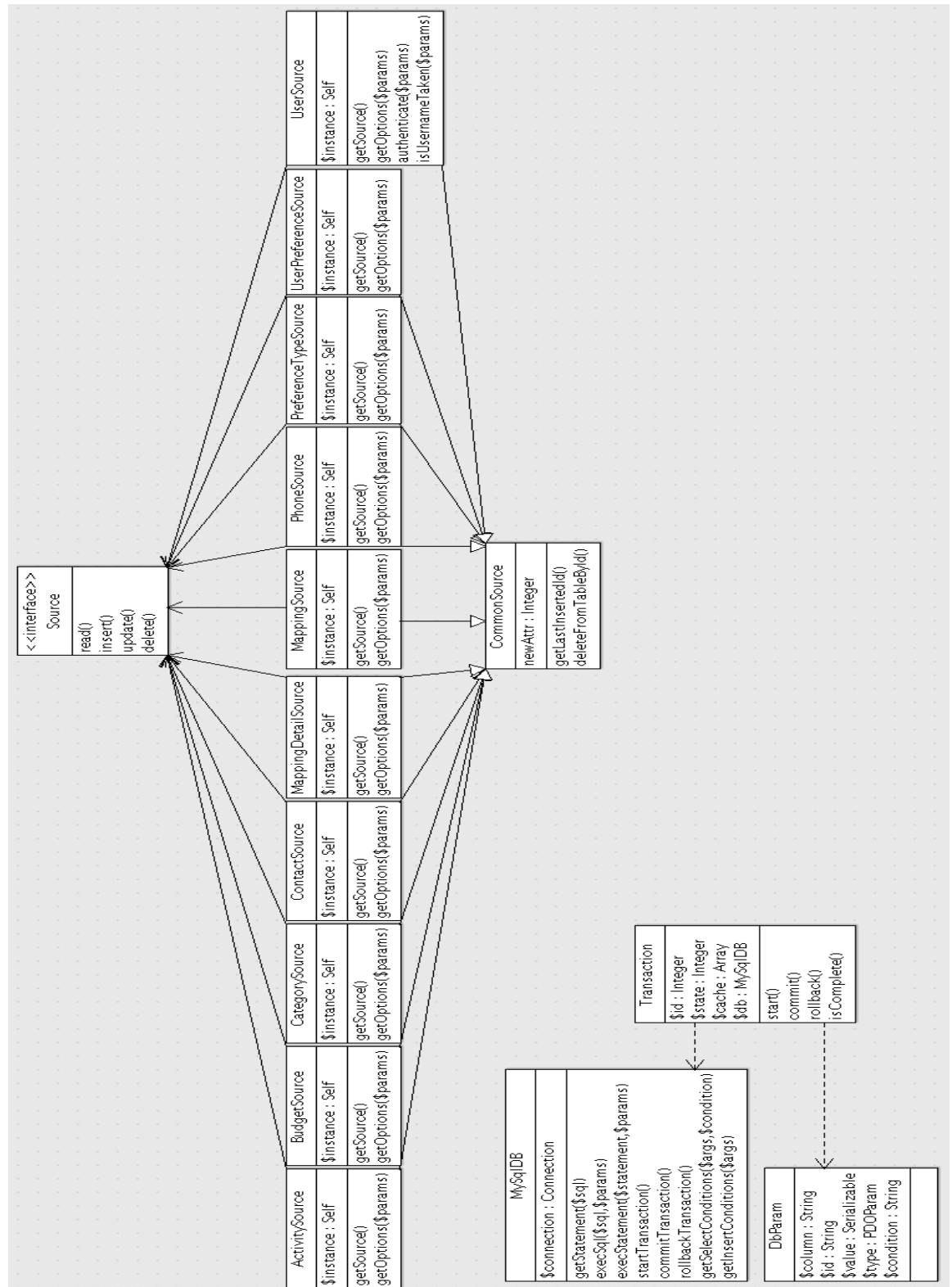
3.2.11 Maintainability

Changing the look and feel of the application can be easily done just by switching the style sheet (ibudget.css). Also by abstracting out common element such as site-wide header and footer, making a change to them makes it a trivial task. Moreover, modular design has been incorporated throughout the whole system, so future requests for enhancements can be easily accommodated.

3.2.12 Performance

The application should load and display the pages in less than 10 seconds on an idle system with no network congestion. In addition, any built-in reports should be displayed in less than 60 seconds as taking longer than that will have a serious impact on user experience.

Appendix: Source Class Diagram



Object Class Diagram

